Universidad Zaragoza
1542

Trabajo Fin de Máster

Técnicas de aprendizaje automático por particionamiento recursivo basado en modelos.

Autor:

Francisco Javier Carela Ferrer

Directores:

José Tomás Alcalá Nalvaiz

Vanesa Cortés Utrillas

Máster Universitario en Modelización e Investigación Matemática, Estadística y Computación

Facultad de Ciencias
Universidad Zaragoza

2020

Agradecimientos

Mis agradecimientos se dirigen, en primer lugar, a Tomás por el apoyo en el proceso de elaboración de todo el trabajo, a pesar de la incertidumbre de los últimos meses, que nos ha hecho ganar en capacidad de adaptación.

También agradecer al equipo de apoyo técnico de la base de datos *Amadeus y* al profesor Luis Ortín por la ayuda en la selección de las variables. Por último, agradecer a la profesora Maribel Ayuda por servirme de guía e inspiración en la elección del trabajo.

Gracias :).

Índice

Índice de figuras

Índice de tablas

Abstract

In recent years, biased and unbiased recursive partitioning algorithms have been applied to datasets from very different scopes, within fields of knowledge such as medicine, engineering or finance. In this work we will try to apply unbiased recursive partitioning to a set European companies listed on the stock exchange.

The unbiased recursive partitioning algorithms (GUIDE, Ctree and MOB) have recently been developed, and they solve some problems of biased recursive partitioning algorithms. More traditional tree algorithms implement exhaustive search procedures to find both, the best split point and split variable in one step by directly comparing all possible split points in all possible split variables. However, it has been shown that this is not only computationally expensive but also biased towards split variables with many possible split points. The unbiased recursive partitioning first selects the covariate using statistical inference, and then, optimizes an objective function to choose the split point.

To assess whether splitting of the node is necessary, a fluctuation test for parameter instability is performed. If there is significant instability with respect to any of the partitioning variables, split the node into B locally optimal segments and repeat the procedure. If no more significant instabilities can be found, the recursion stops and returns a tree where each terminal node (or leaf) is associated with a model, for example a linear or a logit model.

In summary, Model-based recursive partitioning (MOB) is fitted following the next basic steps: first fit a parametric model to a data set, second test for parameter instability over a set of partitioning variables, third if there is some overall parameter instability, split the model with respect to the variable associated with the highest instability, fourth repeat the procedure in each of the daughter nodes.

In this report, the advantages of model-based recursive partitioning (MOB) have been verified. It has also been observed how the estimation varies when the nature of the response variable changes, that is, by having a continuous, binary, or ordinal response variable. It has also been tested how MOB can fit different models in the terminal nodes, such as Logit, Lineal model, or Random Forest.

Resumen

En los últimos años, se han aplicado algoritmos de partición recursiva sesgados e imparciales a datos de diferente naturaleza, dentro de campos como la medicina, la ingeniería o las finanzas. En este trabajo intentaremos aplicar una partición recursiva imparcial a un conjunto de empresas europeas que cotizan en bolsa.

Los algoritmos de partición recursiva insesgados (GUIDE, Ctree y MOB) se han desarrollado recientemente y resuelven algunos problemas de algoritmos de partición recursiva sesgados, como emplear estrategias de búsqueda ambiciosas, comparando directamente todos los puntos de split posibles en todas las covariables disponibles. La partición recursiva insesgada primero selecciona la covariable mediante inferencia estadística y luego optimiza una función objetivo para elegir el punto de split.

Con MOB podemos particionar un conjunto de datos en subconjuntos basándonos en variables de partición. Primero elegimos un modelo paramétrico para el conjunto de datos. Luego realizamos una prueba de inestabilidad de parámetros sobre un conjunto de variables de partición. Si hay alguna inestabilidad general de los parámetros, particionamos el conjunto con respecto a la variable asociada con la mayor inestabilidad. Repetiremos el procedimiento en cada una de las submuestras resultantes.

En un primer capítulo teórico se han presentado las generalidades de los árboles de clasificación y regresión y algunos modelos predictivos básicos. En el segundo capítulo se ha detallado el proceso del algoritmo de particionamiento recursivo con especial atención a los contrastes de hipótesis basados en el proceso empírico de fluctuación que se utiliza para seleccionar variables de segmentación. En el capítulo 3 se ha abordado el análisis de un conjunto de datos, con el objetivo de predecir la rentabilidad de empresas europeas a través de indicadores macroeconómicos y contables. Se completa la memoria con un Glosario de Términos y dos Anexos.

En este trabajo, se han verificado las ventajas de la partición recursiva basada en modelos (MOB). También se ha observado cómo varía la estimación cuando cambia la naturaleza de la variable de respuesta, es decir, al tener una variable de respuesta continua, binaria u ordinal. También se ha analizado cómo MOB puede encajar diferentes modelos en los nodos terminales, como Logit, modelo lineal generalizado o Random Forest.

Motivación

La toma de decisiones ha sido y es, foco de estudio de algunas de las grandes disciplinas intelectuales como las matemáticas, la ingeniería, la psicología, la economía o la política entre muchas otras. Su carácter interdisciplinar permite tanto plantear como abordar un mismo problema desde diversos puntos de vista consiguiendo enriquecer y ampliar su comprensión.

Para plantear y resolver los problemas asociados a la toma de decisión, se recurre a los sistemas de apoyo a la decisión. Una de las definiciones comúnmente más aceptada sobre lo que es un sistema de apoyo a la decisión nos la da [1] :"*Un sistema de apoyo a la decisión es un sistema de información basado en computadora que combina modelos y datos en un intento de resolver semiestructurados y algunos problemas no estructurados con amplia participación del usuario*". En este trabajo se utilizará el software libre R como sistema de apoyo a la decisión para ayudar a ajustar los modelos necesarios.

La motivación de este trabajo surge de la necesidad de aplicar una variedad de técnicas matemáticas y estadísticas, que han sido aplicadas con éxito a datos de diversa naturaleza, a un conjunto de datos concreto, en este caso un conjunto de empresas cotizadas europeas.

Capítulo 1: Introducción

En este trabajo nos centraremos dentro del aprendizaje supervisado, en particionamiento recursivo en arboles de regresión, pero antes haremos una aproximación a los distintos tipos de aprendizaje supervisado.

Para conocer en profundidad los términos que se exponen en la introducción se puede acudir a [2] , en esta introducción únicamente se realizará una breve presentación de las diferentes técnicas, intentando obtener una imagen global de algunas de ellas, y conocer qué lugar ocupa el particionamiento recursivo en los árboles de regresión.

Aunque es difícil encontrar una definición precisa de lo que es aprendizaje automático, podríamos decir que es un campo de la inteligencia artificial, que nos permite a través de distintas técnicas que el ordenador aprenda por sí mismo. Para ello le proporcionamos datos con los que llevará acabo el entrenamiento con el fin de resolver un problema, como puede ser realizar una regresión o una clasificación. Dentro del aprendizaje automático, existen dos grandes ramas: aprendizaje no supervisado y aprendizaje supervisado.

En el aprendizaje no supervisado no tenemos unas etiquetas previas asignadas a los datos del conjunto, y el objetivo será el de encontrar patrones entre los datos, que nos permitan su agrupación en función de las diferentes variables. Algunas de las técnicas que se utilizan en el aprendizaje no supervisado para agrupación son: K-Medias, Estimación de densidades o Vecinos más próximos.

En el aprendizaje supervisado, por el contrario, los datos del conjunto si tienen una etiqueta de entrada asignada, y a través de una función podremos asignar, con cierta probabilidad, una etiqueta de salida. En base al tipo de etiqueta de salida distinguiremos entre técnicas de clasificación, las cuales asignan una etiqueta discreta a los datos, y de regresión, en cuyo caso la etiqueta de salida es una variable continua.

Entre las técnicas de aprendizaje supervisado utilizadas para regresión destacamos: Regresión lineal paramétrica, Regresión no paramétrica (técnicas de suavizado, modelos aditivos), Redes neuronales, Redes funcionales y Árboles de regresión. Algunas de las técnicas de aprendizaje supervisado utilizadas para clasificación son: Análisis discriminante, Modelos lineales y no paramétricos generalizados (Modelos Logit, GAMs), Estimación de densidades, Árboles de clasificación, Máquinas de vector soporte, Redes probabilísticas y Boosting.

1.1 Modelo Logit

El modelo Logit es un tipo de modelo lineal general, donde la variable de regresión Y_i surge del análisis de la probabilidad $\pi_i = P(Y_i = 1|x_i) = G(x_i'\beta)$ siendo x_i el vector de variables explicativas del modelo y $x_i'\beta = \beta_0 + \beta_1 x_1 + \cdots + \beta_p x_p$ el predictor lineal. El modelo Logit o de regresión logística supone una relación lineal entre los odds-ratios $ln\frac{\pi_i}{1-\pi_i}$ en escala logarítmica es del cociente de la probabilidad de que un valor pertenezca a la clase 1, es decir π_i , entre la probabilidad de que un valor pertenezca a la clase 0, es decir $1 - \pi_i$

Normalmente el método más utilizado para estimar los coeficientes β del modelo es la máxima verosimilitud. La cual, para este caso concreto, se obtiene de la siguiente expresión:

$$L(\beta) = \prod_{i=1}^{n} G(x_i'\beta)^{Y_i}(1 - G(x_i'\beta))^{1-Y_i} \qquad (1)$$

con $G(x_i'\beta) = \frac{\exp(x_i'\beta)}{1+\exp(x_i'\beta)}$. La estimación de los parámetros $\hat{\beta}_{ML}$ se obtiene por la máxima verosimilitud , maximizando el logaritmo de la expresión dada en (1), es decir, $l(\beta) = \log(L(\beta))$.

1.2 Modelo logit multinomial

En la línea del modelo logit, también existe el modelo logit multinomial, o de regresión logística multinomial [3]. Este tipo de modelo se forma a partir de variables respuestas categóricas, en los que el orden de estas categorías es irrelevante. Se denota $\{\pi_1, \dots, \pi_J\}$ como la probabilidad de las respectivas categorías respuesta, cumpliéndose $\sum_{j=1}^{J} \pi_j = 1$. Se escoge arbitrariamente una categoría de base o de referencia, a la que se denota (J), y los logits de cada categoría respecto a esta son:

$$\log\left(\frac{\pi_j}{\pi_J}\right), \qquad j = 1, \dots, J-1$$

el modelo logit con una variable predictora x tiene la forma:

$$\log\left(\frac{\pi_j}{\pi_J}\right) = \alpha_j + \beta_j x, \qquad j = 1, \dots, J-1$$

de esta forma el modelo tendrá $J-1$ ecuaciones logit, con parámetros separados para cada una, de tal forma, que un modelo con $J=2$ resultaría simplemente un modelo de regresión logística como el expuesto en el apartado 1.1.

Para estimar la probabilidad de la variable respuesta , tendremos:

$$\pi_j = \frac{\exp(\alpha_j + \beta_j x)}{\sum_h \exp(\alpha_h + \beta_h x) + 1}, \qquad j = 1, \dots, J-1, \quad \pi_J = \frac{1}{1 + \sum_h \exp(\alpha_h + \beta_h x)}$$

1.3 Modelo logit ordinal acumulado

Otra variación al modelo logit, surge cuando las variables respuesta categóricas son ordinales. Se define la probabilidad acumulada de que la variable respuesta Y resulte ser de la categoría j o por debajo, para cada posible j. La probabilidad de la categoría j acumulada se define como:

$$P(Y \leq j) = \pi_1 + \cdots + \pi_j, \qquad j = 1, \dots, J.$$

a través de la probabilidad acumulada se refleja el ordenamiento de las categorías: $P(Y \leq 1) \leq P(Y \leq 2) \cdots \leq P(Y \leq J) = 1$. El modelo no utiliza la probabilidad acumulada final, $P(Y \leq J)$, ya que por definición es igual a 1, por lo que el logit $j-1$ se define como:

$$logit[P(Y \leq j)] = log\left(\frac{P(Y \leq j)}{1 - P(Y \leq j)}\right) = log\left(\frac{\pi_1 + \cdots + \pi_j}{\pi_{j+1} + \cdots + \pi_J}\right), \qquad j = 1, \dots, J-1$$

para este modelo, el predictor X del modelo se introduce como:

$$logit[P(Y \leq j)] = \alpha_j + \beta x, \qquad j = 1, \dots, J-1$$

el parámetro β describe el efecto de X en el logit de las probabilidades de la respuesta en la categoría j o inferior. β no tiene el subíndice j, ya que el modelo asume el mismo efecto de X para todos los niveles de Y. Si tomamos dos valores particulares x_1 y x_2 de X, los odds ratios expresan probabilidad acumulada y sus complementarios:

$$\frac{P(Y \leq j | X = x_2)/P(Y > j | X = x_2)}{P(Y \leq j | X = x_1)/P(Y > j | X = x_1)}$$

el logaritmo de este odd ratio es la diferencia entre el logit acumulado de estos dos valores de x_j, es decir, es igual a $\beta(x_2 - x_1)$, proporcional a la distancia entre los valores de X.

1.4 Árboles de regresión

Como se ha expuesto, nos centraremos en la técnica de particionamiento recursivo en árboles de regresión, la cual se engloba dentro de las técnicas de aprendizaje supervisado. Para ello haremos un acercamiento a los tipos de Árboles de regresión y los distintos términos que nos encontraremos al tratar con ellos.

El objetivo que se persigue cuando se modeliza a través de un árbol de regresión es el de segmentar el conjunto de datos para encontrar grupos homogéneos a partir de una variable respuesta. No es claro el origen de esta técnica. Algunas referencias apuntan al artículo de J.R Quilan de 1986 " *Introduction of decision trees*", sin embargo, para trazar el origen de los árboles de regresión, nos basaremos en [4] .

1.4.1 AID

El primer modelo que puede ser visto como un árbol de regresión o de decisión es el modelo AID (*Automatic Interaction Detection*) presentado en [5] donde ya se diferenció entre variables *ordenadas* si toman valores numéricos y las cuales tienen un orden intrínseco y *categóricas* en otro caso.

Se define concepto de árbol binario, el cual parte de un nodo raíz y de un conjunto X de variables y se selecciona una de ellas para hacer un split del conjunto en dos nodos hijos y seguir el proceso de forma recursiva en cada nodo. Si en X hay una variable ordinal con n categorías se deben analizar $n-1$ posibles particiones. En el caso de tener una variable categórica con m posibles valores o categorías se deben analizar hasta $(2^{m-1}-1)$ posibles particiones.

La relevancia o interés de una partición concreta se mide a través del concepto de impureza en el nodo t. Este método escoge el *split* que minimiza la suma de las impurezas en los dos nodos hijos. La división se detendrá en el momento en el que la reducción de la impureza sea menor que un cierto umbral respecto al nivel de impureza anterior a dicha división o cuando el tamaño del nodo no sea significativo. El valor predicho de Y en cada nodo terminal es la media muestral de dicho nodo y el resultado de aplicar esta técnica es una estimación por partes de una función de regresión.

1.4.2 CHAID

El modelo CHAID propuesto en [5] se centra en árboles con respuesta binaria y variables categóricas. La selección de un split se realiza en base a criterios de significación estadística; contrastando la asociación entre la variable respuesta y las posibles variables de segmentación. La partición se realiza si el contraste es estadísticamente significativo. Este método se basa en el test chi-cuadrado, contrastando la variable respuesta con el predictor y ver, en cada caso, la asociación y si es significativo para más tarde realizar la agrupación.

1.4.3 CART

El método CART propuesto en [6] supone un importante avance, al unificar los criterios para árboles de clasificación y de regresión. Continuando con el trabajo de AID, sustituyen el método de detención, por uno en el que se deja que el árbol se haga más grande. Una vez se tiene el árbol máximo se realiza podado bajo el criterio de escoger aquel tamaño que tenga la estimación de error de validación cruzada más baja. De esta forma se solucionan algunos problemas de *overfitting* y *underfitting* que tenía el método de AID, aunque con un mayor coste de cómputo. Con CART también se pueden utilizar combinaciones lineales de las variables para realizar particiones en los distintos nodos.

El objetivo en esta técnica es el de minimizar el coste del árbol, que viene reflejado en la probabilidad de mala clasificación. Para un árbol de clasificación en cada nodo este coste es:

$$r(t) = 1 - \max_j p(j/t) \tag{2}$$

siendo $p(j/t)$ la probabilidad de que se asigne j en el nodo t. El coste total del árbol será, por lo tanto:

$$R(T) = \frac{\sum_{t\in\tilde{T}} p(t)r(t)}{r(root)} * 100 \tag{3}$$

siendo $\tilde{T}$ el conjunto de nodos terminales en T, $p(t)$ la probabilidad de asignación del nodo t y $r(root)$ el coste de mala clasificación en el nodo raíz.

Otras medidas de impureza comúnmente utilizadas para los árboles de clasificación serían Gini, Entropía o Twoing [7] . Para un árbol de regresión con CART , el coste de mala clasificación

total vendrá dado por la suma del error de predicción al cuadrado entre el número de nodos, es decir, la varianza.

$$r(t) = \frac{1}{n_t}\sum_{i=1}^{n_t}(y_{it} - \bar{y}_t)^2 \tag{4}$$

el valor y_{it} es la observación i asociada al nodo t, $\bar{y}_t$ es la predicción asociada al nodo t , También se podría utilizar la desviación absoluta como medida de impureza para el caso de los árboles de regresión.

Una vez tenemos seleccionado un criterio de medida de la impureza, tendremos que seleccionar cual será la selección de impureza óptima para realizar la poda. Para podar el árbol en CART se impondrá una penalización según la complejidad a través de:

$$Min(R(T) + \alpha|T|) \tag{5}$$

pudiéndose interpretar $\alpha = \frac{R(t)-R(T_t)}{|T_t|-1}$ como cantidad que se debe penalizar por el tamaño del árbol para que no valga la pena continuar haciendo divisiones a partir de ese nodo, $R(T_t) + \alpha|T_t|$ es el coste del subárbol descendiente del nodo t penalizado por su tamaño y $R(T_t) + \alpha$ es el coste del árbol después de haber podado su subárbol descendiente. Para llevar a cabo este proceso se divide la muestra en muestra de entrenamiento y de validación.

También se puede llevar a cabo un proceso de validación cruzada en el que se dividirá al azar la muestra total en k partes y se formarán árboles máximos con $n, n - n_1, n - n_2, \ldots$ individuos. A partir de (3) se calculan los valores de impureza para cada valor α:

$$R^{vc}(T_\alpha) = \sum_{l=1}^{k}\frac{\sum_{t\in\tilde{T}_\alpha} p_l(t) r_l^{vc}(t)}{r_l^{vc}(root)} * \frac{100}{k} \tag{6}$$

y teniendo como referencia de selección de árbol óptimo (5) para este caso:

$$Min\left(R^{vc}(T_\alpha) + 1 \times desv\ tip\left(R^{vc}(T_\alpha)\right)\right) \tag{7}$$

las técnicas AID CHAID y CART comentadas hasta ahora emplean directamente todos los posibles puntos de split de las variables, sin embargo, la investigación posterior [8] ha demostrado

que esto está sesgado hacia la selección de aquellas variables que tienen mayores puntos de *split* potenciales. Por lo tanto, recientemente han surgido otro tipo de algoritmos de particionamiento recursivo insesgado que, en base a la inferencia estadística utilizan test de fluctuación para la inestabilidad de los parámetros.

1.4.4 Random Forest

Terminamos esta sección introduciendo una variante muy popular de algoritmo de clasificación. Ha sido desarrollado en [9] y es un algoritmo de tipo aprendizaje por consenso que utiliza los árboles de regresión para realizar las predicciones, y clasifica según la categoría más votada, para llegar a un resultado por consenso calculando la media de todos ellos.

Random Forest selecciona submuestras de variables aleatorias con reemplazamiento. Esta novedad permite, en cada split, que los predictores hayan sido elegidos de manera aleatoria, haciendo que los árboles calculados no estén correlacionados. Los árboles que se calculan se dejan crecer limitando la profundidad del árbol, sin realizar podado. Este tipo de algoritmo puede utilizarse tanto para clasificación como para regresión.

Capítulo 2: Particionamiento recursivo insesgado

En este apartado vamos a comentar brevemente los distintos algoritmos de particionamiento recursivo insesgado, basándonos en [8] . El objetivo del particionamiento recursivo sigue siendo el de encontrar la mejor variable para particionar dentro de las distintas covariables, pero incorporando ahora elementos de inferencia estadística, a través de distintos test midiendo el p-valor para encontrar puntos de split. De forma resumida y antes de presentarlos con detalle, los métodos que se vamos a introducir se diferencian en que **GUIDE** utiliza pruebas de significatividad estadística a través independencia de los residuos, **CTree** utiliza test de permutación en un marco de inferencia condicional y **MOB** utiliza test de fluctuación basado en el Teorema Central del Límite para los parámetros estimados.

2.1 GUIDE

El método GUIDE introducido por [10] desarrolla el modelo de clasificación y regresión de árboles GUIDE .Combina en un único modelo los árboles recursivos y modelos lineales, pudiendo obtener los árboles de clasificación y regresión como casos especiales. En cada nodo se ajusta un modelo de regresión, aunque también se puede ajustar un valor constante tal que $\hat{\beta} = \bar{y} = \frac{1}{N}\sum_{i=1}^{N} y_i$. Los test que seleccionan las variables de split se basan en los residuales de una regresión lineal simple, tal que:

$$r(Y, X, \hat{\beta}) = Y - \hat{\beta}_0 - \hat{\beta}_1 \cdot X \tag{8}$$

en este algoritmo es necesario que los residuales sean dicotomizados (8), diferenciándolos entre positivos y negativos, y que la variable de split sea categorizada, a menos que esta variable sea ya categórica, se divide en cuatro categorías de acuerdo con sus cuartiles. Después se realiza un test χ^2 para comprobar la independencia de los residuos dicotomizados con respecto a cada una de las variables categorizadas, es decir, aquellas que utilizamos para el split. Una vez hemos elegido la variable split, escogemos el punto de split que minimiza el criterio de bondad de ajuste que hayamos seleccionado.

2.2 Ctree

Hothron desarrollo en [11] donde trabaja este tipo de árboles más clásicos, basados en la idea de proporcionar una regresión no paramétrica del árbol de regresión en un marco de inferencia condicional aplicando un test de permutación.

Para seleccionar la variable split se realizar un test que relaciona la variable respuesta con la posible variable de split $g(Z_j), j = 1, \dots, J$. En el caso de que se ajuste un modelo paramétrico para la variable respuesta Y con algunas covariables X empleadas como regresores de la variable, se puede utilizar una transformación basada en el modelo $h(Y) = s(Y, X, \hat{\beta})$, utilizando por ejemplo los residuales de un modelo lineal con coeficientes β o la función score en el caso de un modelo con función objetivo ℓ:

$$s(Y, X, \beta) = \frac{\partial \ell(Y, X, \beta)}{\partial \beta} \tag{9}$$

la decisión de usar Z como variable de split se basa ahora en utilizar un test de independencia o no asociación lineal entre $h(Y)$ y $g(Z)$.

Por ejemplo, en una situación básica donde $h(Y) = Y$ y $g(Z) = Z$, el contraste estaría basado en el coeficiente de correlación entre variables. Una correlación o asociación lineal fuerte indicaría la conveniencia de utilizar Z como variable de particionamiento. Con otro tipo de variables se podría considerar pruebas del tipo ANOVA o pruebas del tipo Chi-cuadrado, siempre que permitan medir el grado de asociación.

2.3 MOB

Para el desarrollo de este apartado se ha usado como referencia [12] . Vamos a considerar un modelo paramétrico $\mathcal{M}(Y, \theta)$con observaciones $Y \in \mathcal{Y}$ y un vector de dimensión k de los parámetros $\theta \in \Theta$. Dadas n observaciones $Y_i (i = 1, \dots, n)$ el modelo puede ser ajustado minimizando la función objetivo $\Psi(Y, \theta)$, obteniendo el parámetro estimado $\hat{\theta}$:

$$\hat{\theta} = \underset{\theta \in \Theta}{\operatorname{argmin}} \sum_{i=1}^{n} \Psi(Y_i, \theta) \tag{10}$$

las dos técnicas más populares para llevar a cabo esta estimación son los mínimos cuadrados ordinarios o la máxima verosimilitud, aunque también es posible aplicar otros. Para el caso de los mínimos cuadrados la función objetivo Ψ a minimizar es la suma de los residuos al cuadrado, es decir la suma de la diferencia al cuadrado del valor observado y el valor estimado, mientras que para el caso de la máxima verosimilitud Ψ es el valor negativo del logaritmo de la verosimilitud.

Muchas veces no será razonable asumir que un modelo global $\mathcal{M}(Y, \theta)$ajusta las n observaciones correctamente, pero sin embargo si puede ser posible dividir las observaciones con respecto a algunas covariables de modo que se pueda encontrar un modelo que se ajuste bien localmente en cada uno de los segmentos de la partición . En ese caso tendremos P variables de particionamiento $Z_j \in \mathcal{Z}_j$ $(j = 1, \ldots, P)$ para adaptativamente encontrar una buena aproximación de esta partición. Asumiremos que la partición $\{\mathcal{B}_b\}_{b=1,\ldots,B}$ del espacio $\mathcal{Z} = \mathcal{Z}_1 \times \cdots \times \mathcal{Z}_P$ posee B segmentos, de tal forma que en cada uno de los segmentos $\mathcal{B}_b$ tendremos un modelo $\mathcal{M}(Y, \theta_b)$ asociado a ese segmento con un parámetro específico θ_b. Denotaremos este modelo segmentado como $\mathcal{M}_b(Y, \{\theta_b\})$ donde $\{\theta_b\}_{b=1,\ldots,B}$ es la combinación completa del parámetro.

Si tenemos la partición correcta de $\{\mathcal{B}_b\}$ la estimación de los parámetros $\{\theta_b\}$ que minimizan la función objetivo global puede obtenerse calculando el óptimo local del parámetro estimado $\hat{\theta}_b$ en el segmento $\mathcal{B}_b$. Cuando$\{\mathcal{B}_b\}$ no es conocido, se puede expresar a partir de (10) la minimización de la función objetivo global como :

$$\sum_{b=1}^{B} \sum_{i \in I_b} \Psi(Y_i, \theta_b) \tag{11}$$

sobre todas las particiones posibles $\{\mathcal{B}_b\}$ es más complejo, aunque el número de segmentos B es fijo. En el caso de que haya más de una variable de partición, es decir $P > 1$ el número de particiones potenciales es demasiado alto. Además, si el número de segmentos B no es conocido es aún más complicado , aunque es cierto que se pueden eliminar algunas particiones que no serán de utilidad , como aquellas en las que cada observación es su propio segmento, por lo que se requerirá un tamaño mínimo a cada segmento. En el caso de que haya una sola variable de partición $(P = 1)$ el punto de división óptimo es más sencillo de obtener a través del análisis del cambio estructural a través de test como CUSUM o CUSUM-sq.

El algoritmo lo podremos describir en los siguientes pasos:

1. A través de la minimización de la función objetivo (10) obtenemos $\hat{\theta}$ en el nodo actual para todas las observaciones.

2. Evaluamos la estabilidad del parámetro estimado con respecto a todas las$Z_1, \cdots, Z_P$. Si existe alguna inestabilidad seleccionar la variable Z_j asociada con la mayor inestabilidad de los parámetros, en otro caso detenerse.
3. Calcular el punto de split que minimiza localmente Ψ, ya sea para un número fijo o adaptativo de splits.
4. Dividir el nodo en nodos hijos y repetir el proceso.

Estimación de los parámetros

En condiciones de estabilidad moderada [13] $\hat{\theta}$ se obtiene de resolver las condiciones de primer orden:

$$\sum_{i=1}^{n} \psi\left(Y_i, \hat{\theta}\right) = 0 \tag{12}$$

donde $\psi(Y, \theta) = \frac{\partial \Psi(Y,\theta)}{\partial \theta}$ es el score relativo de $\Psi(Y, \theta)$. El score de la función en los parámetros estimados $\psi_i = \psi\left(Y_i, \hat{\theta}\right)$ se utilizará más tarde para detectar desviaciones sistemáticas de su valor medio 0.

Contraste de inestabilidad de los parámetros

En este punto, el objetivo es averiguar si los parámetros del modelo son estables sobre cada ordenación o si particionando la muestra con respecto a uno de los Z_j es capaz de recoger inestabilidades en los parámetros y así mejora el ajuste.

Para evaluar la inestabilidad del ajuste, una idea es revisar si los scores de $\hat{\psi}_i$ fluctúan aleatoriamente con media 0 o tienen desviaciones sistemáticas de 0 en relación con el orden marcado por Z_j . Las desviaciones pueden ser medidas mediante la definición de un proceso empírico de fluctuación:

$$W_j(t) = \hat{J}^{-1/2} n^{-1/2} \sum_{i=1}^{\lfloor nt \rfloor} \hat{\psi}_{\sigma(Z_{ij})} \qquad (0 \leq t \leq 1) \tag{13}$$

donde $W_j(t)$ es el proceso suma parcial de los scores ordenados por la variable Z_j escalado por el número de observaciones n , y la estimación adecuada $\hat{J}$ de la matriz de varianzas-covarianzas $\mathrm{COV}\left(\psi(Y,\hat{\theta})\right)$, por ejemplo, $\hat{J} = n^{-1}\sum_{i=1}^{n}\psi(Y_i,\hat{\theta})\psi(Y_i,\hat{\theta})^{\mathrm{T}}$. También se pueden utilizar otros estimadores robustos frente a heterocedasticidad o frente a heterocedasticidad y autocorrelación. $\sigma(Z_{ij})$ denota una permutación de ordenación que devuelve el valor *antirank* de la observación Z_{ij} en el vector $Z_j = (Z_{1j},\ldots,Z_{nj})^{\mathrm{T}}$.

Bajo la hipótesis nula de estabilidad de los parámetros, este proceso de fluctuación empírica está sujeto al Teorema Central del Límite, y converge a un puente Browniano W^0.

Este contraste de estabilidad de parámetros cae dentro de una clase más general de contrastes denominada "*generalized M-fluctuation test"* estudiados en [14]. Incluye algunos test de cambio estructural basados en CUSUM, test basados en el score, o multiplicadores de Lagrange entre algunos otros.

Para implementar el algoritmo de particionamiento recursivo podrían utilizarse cualquiera de los test nombrados anteriormente, aunque hay dos especialmente interesantes para evaluar las variables de particionamiento numéricas y categóricas Z_j respectivamente.

Evaluación de variables numéricas

Para el caso de variables de particionamiento de tipo continuo el funcional se define como:

$$\lambda_{\sup LM}(W_j) = \max_{i=\underline{i},\ldots,\bar{\imath}}\left(\frac{i}{n}\cdot\frac{n-i}{n}\right)^{-1}\left\|W_j\left(\frac{i}{n}\right)\right\|_2^2 \qquad (14)$$

este funcional se puede escribir como el máximo de la norma L_2 al cuadrado del proceso de fluctuación empírica escalado por su función de varianza. Este estadístico es equivalente a un estadístico del tipo supLM, es decir, supremo de estadísticos del tipo de Multiplicadores de Lagrange, utilizando habitualmente contra una alternativa de un único punto de cambio potencial; y donde dicho cambio se produce en el intervalo $[\underline{i},\bar{\imath}]$ donde $\underline{i}$ es un segmento de tamaño mínimo requerido y $\bar{\imath} = n - \underline{i}$.

Este test es asintóticamente equivalente al test de Chow [15] , pero con la ventaja de que el modelo solo tiene que ajustarse una vez bajo la hipótesis nula de estabilidad de los parámetros, y no para cada punto de cambio estructural posible.

Evaluación de variables categóricas

Para recoger la inestabilidad con respecto a la variable categórica Z_j con C categorías diferentes, se necesita un estadístico diferente, porque por definición, Z_j tiene empates o coincidencias entre sí y por lo tanto no es posible un ordenamiento total de las observaciones. El estadístico que vamos a definir es insensible a la agrupación de los niveles de C y al del grupo de observaciones:

$$\lambda_{\chi^2}(W_j) = \sum_{c=1}^{C} \frac{|I_c|^{-1}}{n} \left\| \Delta_{I_c} W_j \left(\frac{i}{n} \right) \right\|_2^2 \tag{15}$$

donde $\Delta_{I_c} W_j$ es el incremento del proceso empírico de fluctuación sobre las observaciones en la categoría $c = 1, \ldots, C$, es decir la suma de los scores en la categoría c. El estadístico del contraste se puede ver como la suma ponderada de la norma L_2 al cuadrado de los incrementos, los cuales tienen una distribución χ^2 con $k(C-1)$ grados de libertad, a partir de los cuales el p-valor p_j puede ser calculado.

La ventaja de usar este enfoque en el proceso de empírico de fluctuación (13) con los funcionales definidos en (14) y (15) es que los parámetros estimados y los scores solo tienen que ser calculados una vez por nodo. Para realizar los test de inestabilidad de los parámetros, los scores solo tienen que ser reordenados y agregados en un test escalar una sola vez.

Splitting

En este paso del algoritmo, el modelo ajustado tiene que ser dividido con respecto a la variable Z_{j^*} en un modelo segmentado con B segmentos donde B puede ser fijo o determinado de forma adaptativa. Para un número fijo de splits, dos segmentaciones pueden compararse a través de sus funciones objetivo (11) correspondientes. Realizar una búsqueda exhaustiva de las posibles particiones con B segmentos garantiza encontrar una partición óptima, pero puede ser algo costoso, por lo que se presentan dos métodos para encontrar el número de particiones dependiendo de si se trata de variables numéricas o categóricas.

Splitting para variables numéricas

Una búsqueda para un split de $B = 2$ segmentos es factible en $O(n)$ operaciones. Para $B > 2$ una búsqueda exhaustiva puede ser de un orden $O(n^{B-1})$, sin embargo, la partición óptima puede ser encontrada utilizando programación dinámica con una aproximación de $O(n^2)$. Si B es adaptativo podríamos usar criterios de información si los parámetros son estimados por máxima verosimilitud.

Splitting para variables categóricas

Para variables categóricas el número de segmentos no puede ser mayor que el número de categorías $(B \leq C)$. Dos posibles enfoques son siempre dividir en todos los niveles posibles, $(B = C)$ o dividir en el número mínimo de segmentos $(B = 2)$. En el segundo caso la búsqueda del número óptimo de particiones sería del orden de $O(2^{C-1})$. También el criterio de información puede ser una opción en el caso querer determinar de forma adaptativa el número de splits

En [16] se explica cómo recientemente se han combinado los Random Forest con el particionamiento recursivo insesgado, de manera que se considera un subconjunto aleatorio de variables predictoras para cada split. En el siguiente capítulo, se combinará Random Forest con particionamiento recursivo a través de los paquetes `mob` y `mobForest`

Capítulo 3: Aplicación a empresas cotizadas europeas

En este capítulo vamos a llevar a cabo una implementación de la técnica de particionamiento recursivo insesgado de modelos de regresión en sentido general; para ello haremos uso de las técnicas descritas en el capítulo anterior, fundamentalmente del método MOB descrito en el apartado 2.3.

3.1 Elección del conjunto de datos

Utilizaremos un conjunto de datos extraídos de la base de datos *Amadeus* [17]. Consideraremos datos de las empresas cotizadas europeas del año 2017, ordenadas por activos totales. Los datos de cada empresa se referirán, por una parte, a los indicadores contables particulares de cada empresa, como el apalancamiento o el margen operativo, y por otra parte datos macroeconómicos del país en el que cotiza, como son el crecimiento porcentual del PIB o la tasa de desempleo. El proceso que se ha llevado a cabo para la recopilación de la base de datos ha sido el siguiente:

1. **Elección de una base de datos** que nos proporcionase un conjunto de datos fiable a la par que completo. La base de datos *Amadeus*, contiene información completa en torno a 21 millones de empresas en Europa. La información que aporta puede utilizarse para la investigación de empresas individuales, búsqueda de perfiles específicos de empresas o el análisis económico.

2. **Elección del conjunto:** Si bien la población serán las empresas cotizadas europeas, la muestra será aquellas empresas que fueron más grandes de Europa del año 2017, tomando como medida de tamaño el activo total. Otro indicador que se podría tener en cuenta para medir el tamaño de las empresas es el número de trabajadores. De aquí seleccionamos un total de 21.739 observaciones.

3. **Primer filtrado:** Seleccionamos únicamente aquellas que tengan precio de cotización en los años 2016 y 2017, es decir, escogemos únicamente las empresas más grandes cotizadas, eliminando por lo tanto sucursales y empresas no cotizadas. El nuevo conjunto cuenta con 1.539 observaciones. Algunas observaciones tienen valor 0 cuando realmente debe ser un NA, por lo tanto, cambiamos estos valores a NA, utilizando la función de filtrado de Excel para cada una de las columnas.

4. **Elección de variables:** Después se ha realizado la selección de indicadores que podrían afectar directa o indirectamente a la rentabilidad que tiene una acción en el periodo de un año, medido como la tasa de variación en ese año.
   Estos indicadores se dividen entre fundamentales-contables y macroeconómicos. Se incluyen también 2 variables categóricas: País en el que opera (*Country*) e industria a la que pertenece (*Industry*). Se recoge un listado detallado con la definición de cada variable en el Glosario de Términos.

5. **Ajuste y cálculo de variables:** Calculamos algunas variables que nos serán de interés:
   - Tasas de variación del precio de cada acción entre 2016 y 2017. **Esta será la variable a explicar.**
   - Desviación típica de cada acción calculándola a partir de la cotización mensual de cada empresa durante el año 2017. Nos servirá como medida de volatilidad.
   - Cambio en el nombre de algunas variables (Total assets th EUR 2017->*AsstT*, Cash Flow th EUR 2017 -> *CashF*,…etc). También abreviatura de los nombres de la variable industria, medidas con letras a través del Nace Rev, indicador que se explica en detalle en el anexo. Todo esto nos ayudará con la exploración en R.

Finalmente, nuestro conjunto contará con un total de 30 variables y 1.564 observaciones, correspondientes a las empresas europeas cotizadas más grandes de Europa en el año 2017. El objetivo del estudio es la segmentación de las empresas a través de su rentabilidad, para poder realizar un ajuste de un modelo más eficiente.

3.2 Exploración descriptiva y depuración

Para realizar este apartado se ha tenido en cuenta las recomendaciones propuestas en [18]. La exploración se ha implementado principalmente a través del paquete *caret*.

El esquema que vamos a seguir para realizar el análisis será el siguiente:

1. Descripción del tipo de variables
2. Análisis de datos ausentes
3. Distribución de las variables
   3.1 Variable respuesta
   3.2 Variables categóricas
      3.2.1 Agrupación de la variable *Country*

3.3 Variables continuas

4. Análisis preliminar
   4.1 Imputación de valores ausentes
   4.2 Varianza próxima a cero
   4.3 Análisis de correlación
   4.4 Identificación y tratamiento de datos atípicos

Como los puntos tres primeros puntos se centran en un análisis puramente descriptivo, se ha optado por incluirlos en el Anexo. Para muchas de las definiciones expuestas en ese punto, se han tenido en cuenta las que se disponen en [17] y [19]. Se puede encontrar su definición detallada en el Glosario de Términos.

Es necesario indicar que entre los primeros pasos se ha realizado un análisis clúster para la variable *Country*, y trabajaremos con su versión agrupada en 4 categorías.

También crearemos una variable binaria cambiando la variable *Perf_2017*, separando los valores positivos. Además, crearemos 2 variables categóricas a partir de *Perf_2017* con 3 y 5 categorías respectivamente. Con este tipo de variable ajustaremos un modelo Multinomial para cada uno de los nodos terminales.

Puede verse este proceso en el punto 3.2.1 del Anexo.

4. Análisis preliminar

4.1 Imputación de valores ausentes

Vamos a tratar la imputación de valores ausentes. Para ello, acudiremos a la librería *mice*. Primero, se puede consultar en el apartado 2 del anexo. , tenemos algunas variables con más del 10% de datos perdidos, por lo que los eliminaremos de nuestro conjunto. el resultado de nuestro conjunto será:

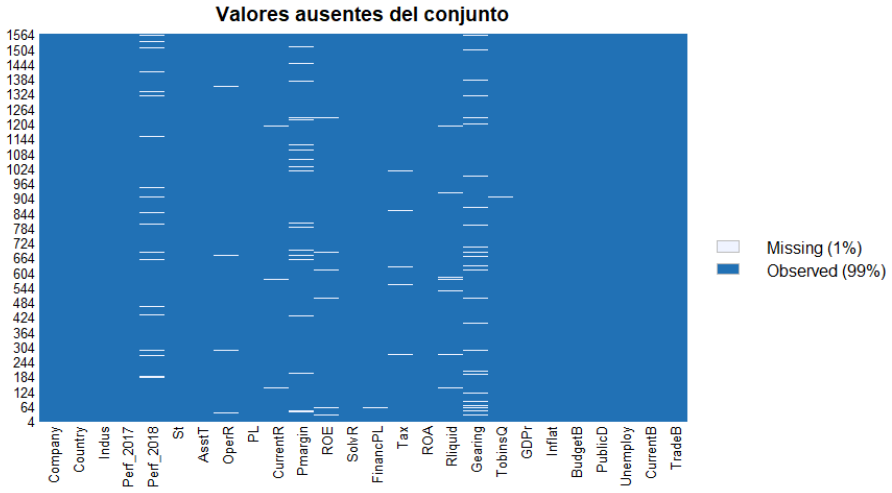


Figura 1: Datos perdidos

se aprecia que hay valores perdidos distribuidos de manera "aleatoria" por lo que ya podremos utilizar las funciones de la librería *mice*. Utilizaremos el método de la media, para así poder completar todo el conjunto.

4.2 Varianza próxima a cero

En este apartado comprobaremos si hay variables con varianza próxima a cero, y las eliminaremos en ese caso. Esto se realiza porque esas variables no nos aportarán apenas información cuando ajustemos el modelo. Vamos a utilizar la función *nearZeroVar()* del paquete *caret*:

```
 freqRatio percentUnique zeroVar   nzv
AsstT     2.000000     99.936061   FALSE FALSE
OperR     2.000000     98.593350   FALSE FALSE
PL        1.000000     99.232737   FALSE FALSE
CurrentR  1.000000     78.772379   FALSE FALSE
Pmargin   1.000000     92.902813   FALSE FALSE
ROE       1.500000     96.035806   FALSE FALSE
SolvR     1.000000     98.657289   FALSE FALSE
FinancPL  3.000000     96.994885   FALSE FALSE
Tax       2.000000     96.035806   FALSE FALSE
ROA       1.000000     96.035806   FALSE FALSE
Rliquid   1.000000     71.739130   FALSE FALSE
Gearing   9.000000     91.751918   FALSE FALSE
TobinsQ   1.000000     69.117647   FALSE FALSE
GDPr      1.761194      2.301790   FALSE FALSE
Inflat    1.761194      2.301790   FALSE FALSE
BudgetB   1.761194      2.301790   FALSE FALSE
PublicD   1.761194      2.301790   FALSE FALSE
Unemploy  1.761194      2.237852   FALSE FALSE
CurrentB  1.761194      2.301790   FALSE FALSE
TradeB    1.761194      2.301790   FALSE FALSE
```

R-Output 1

vemos que ninguna variable tiene varianza próxima a cero por lo que no será necesario hacer modificaciones del conjunto en este punto

.

4.3 Análisis de la correlación

Para este apartado utilizaremos la función *FindCorrelation*() para identificar aquellas variables que se encuentre correladas y eliminarlas. El umbral para eliminar estas variables será de 0.75.

Bajo este criterio, eliminamos las variables *OperR, PL, CurrentR y ROA*. Muchas de las variables que se encontraban correladas eran justamente aquellas que tenían un porcentaje de datos perdidos mayor al 10%, por lo que ya se había optado por eliminarlas en el apartado anterior.

4.4 Identificación y tratamiento de datos atípicos

En este apartado, nos basaremos en lo comentado en la parte de análisis exploratorio, en la que visualmente teníamos presencia de outliers, observado a través del diagrama de caja o *boxplot.* Por ejemplo, para la variable respuesta, tenemos:

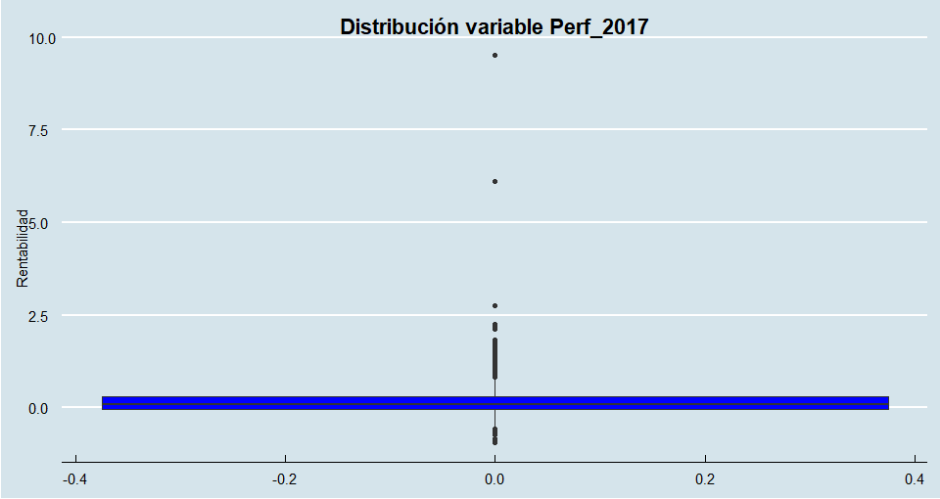


Figura 2: Distribución variable rentabilidad con outliers

vemos que hay valores bastante dispares, por lo que sería recomendable realizar un tratamiento de datos atípicos para que los modelos que ajustemos más tarde tengan **unos estimadores más robustos.**

Optaremos por identificar cuáles son los valores que se encuentran fuera del rango intercuartílico, y reemplazar aquellas observaciones que se encuentran por debajo del límite inferior, por el valor del percentil 10, y aquellas que se encuentran por encima del límite superior, reemplazarlas por el valor del percentil 90. De esta manera obtendremos una distribución algo más manejable:

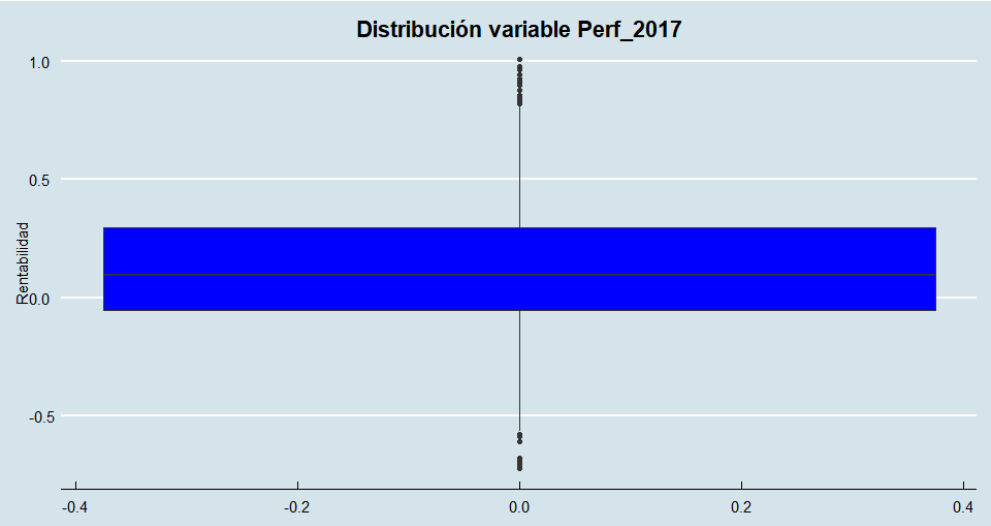


Figura 3: Distribución variable rentabilidad sin outliers

un proceso similar se ha realizado en las variables cuantitativas, por ejemplo, ratios, que solo posean valores superiores a 0, identificando el límite superior del rango intercuartílico, y sustituyéndolo por el valor del percentil 90. El detalle de este proceso se encuentra en el Anexo.

3.3 Ajuste del modelo: Modelo basado en particionamiento recursivo

En este apartado vamos a realizar una aplicación práctica de lo expuesto en el capítulo 3 sobre nuestro conjunto. Para realizar este análisis se ha tomado como referencia [20] . Nos vamos a basar en las librerías **party** y **mob**, aunque también implantaremos **locClass** y **mobForest** .

Para ajustar el modelo utilizaremos la función `mob()` la cual se encuentra dentro del paquete **party.** La función `mob()` tiene los siguientes argumentos:

```
mob(formula, weights, data = list(), na.action = na.omit, model =
glinearModel,control = mob_control(), ...)
```

Script de R. 1

Formula será del tipo `y ~ x1 + ... + xk | z1 + ... + zl` donde las variables a la izquierda de | serán las que explican la variable respuesta `y` mientras que las variables de la derecha son las que se utilizan para el particionamiento. Es posible que haya una misma variable a ambos lados de | .

El valor `model` es el tipo de modelo que ajustará en los nodos terminales. Este parámetro tomará distintos valores:

- `model =linearModel:` ajusta modelos lineales generalizados en los nodos terminales.
- `model = glinearModel` y `family = binomial():` utilizado por mob por defecto , ajusta modelos logit en los nodos terminales.
- `model = multinomModel` : función de ajuste que llamaremos desde la librería **locClass**. Nos permitirá ajustar modelos multinomiales.

También es necesario fijar los parámetros de **mob_control():**

- `minsplit = :`indica el número mínimo de observaciones en cada nodo. Hemos fijado este parámetro en 80.
- `alpha =:` que es el valor que se tendrá en cuenta para el test de fluctuación de las variables ($\alpha = 0.05$ `por defecto`) para rechazar o no la hipótesis nula.
- `objfun =:` el método utilizado para minimizar la función objetivo ajustada en cada nodo `( deviance o logLik).`

Hay que tener en cuenta que cuando cambia el tipo de variable respuesta, también lo hará la relación de las variables dependientes con esta a la hora de realizar los particionamientos, de forma que una variable que puede ser muy determinante para particionar el conjunto en un modelo logit, podría no serlo tanto en un modelo multinomial.

Los criterios que se han elegido para comparar la eficacia de los modelos son los mismos usados en [11], para los que los objetos del paquete **`party`** son sencillos de obtener, y sobradamente conocidos en la literatura estadística:

Logaritmo de la máxima verosimilitud: Consiste en proporcionar la estimación que otorgue máxima probabilidad a los datos observados. La función de verosimilitud se define como:

$$\mathcal{L}(\hat{\theta}) = \sum_{i=1}^{n} \log p(y_i; \hat{\theta})$$

AIC (Criterio de información de Akaike) es un método fácil de interpretar y que se obtiene de la siguiente expresión:

$$AIC = -2\log\left(\mathcal{L}(\hat{\theta})\right) + 2K$$

siendo $\log\left(\mathcal{L}(\hat{\theta})\right)$ el logaritmo de la máxima verosimilitud para el modelo estimado ,valor que también incluiremos como criterio dentro de las tablas, mientras que K es el número de parámetros del modelo. El criterio para escoger un modelo mediante AIC es el de escoger aquel modelo con un valor mínimo, pero el valor de AIC en sí mismo no nos dice nada. Este criterio es útil a la hora de comparar su valor entre diferentes modelos.

MSE (Media de los residuos al cuadrado) es un método que relaciona los valores observados y los estimados por el modelo a través de la siguiente fórmula:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

siendo y_i el valor de la observación i, $\hat{y}_i$ el valor estimado por nuestro modelo para la observación i y n el número de observaciones de nuestro conjunto. Cuanto menor sea el valor de MSE más pequeña será la diferencia entre los valores observados y estimados por nuestro modelo, por lo tanto, será un modelo más preciso.

También incluiremos la **Deviance**, que para el modelo lineal generalizado que se define como:

$$D = -2\left[\log\left(\mathcal{L}(\hat{\theta})\right) - \log(\mathcal{L}_s)\right]\phi$$

siendo $\log\left(\mathcal{L}(\hat{\theta})\right)$ el logaritmo de la máxima verosimilitud para el modelo estimado y $\log(\mathcal{L}_s)$ la estimación de un modelo en el que cada observación tiene su propia media y siendo ϕ un parámetro de escala que mide la dispersión asociada al modelo.

El **tiempo CPU** es el tiempo de ejecución de cada uno de los modelos, y en todas las tablas del trabajo está medido en **segundos**.

Una vez tenemos definidos los valores de `mob_control()`tenemos que definir el valor de `formula` para cada uno de los modelos el valor que queremos coger de la variable respuesta, y también tenemos que definir qué tipo de modelo queremos que nos ajuste. Esto lo fijaremos en `model =` y en `family =` argumentos de la librería **`stats.`**

Para la primera prueba queremos ilustrar como será la salida de uno de estos modelos, para ello ajustamos una fórmula simple que permitirá su representación gráfica en forma de árbol.

```
Perf_2017 ~ Perf_2017_cuant ~~Country + AsstT + Pmargin + St +
Rliquid| Country + Indus + St + AsstT + Pmargin + ROE + SolvR +
FinancPL + Tax + Rliquid + Gearing + TobinsQ +  GDPr +  Inflat +
BudgetB + PublicD + Unemploy + CurrentB + TradeB

mob_control(alpha = 0.05, bonferroni = TRUE, minsplit = 80, trim =
0.1,breakties = FALSE, parm = NULL, verbose = TRUE, objfun = deviance)
```

Script de R. 2: Función mob_control() y parámetros de la función

Modelamos el rendimiento en función de 5 variables que a priori, podrían tener cierta relación con el rendimiento.

Todos los modelos en las tablas que tengan la etiqueta GLM con mob y GLM sin mob tienen como variable respuesta *Perf_2017_cuant*, que es una variable continua, mientras que los modelos Logit con mob y Logit sin mob tienen como variable respuesta *Perf_2017_bi*, que es binaria.

Tabla 1: Ajuste de modelos con mob

Modelos	MSE	Deviance	LogLik	AIC	Tiempo CPU
GLM sin mob	0.0778	97.402	-178.287	370.574	1.30
GLM con mob	0.066	82.715	-40.763	157.526	2.98

Como vemos, tanto el modelo lineal como el Logit han reducido todos los indicadores con la implementación del particionamiento recursivo, por lo que podemos decir que la implementación del particionamiento recursivo en nuestro conjunto ha mejorado nuestra estimación.

La representación del GLM con mob será:

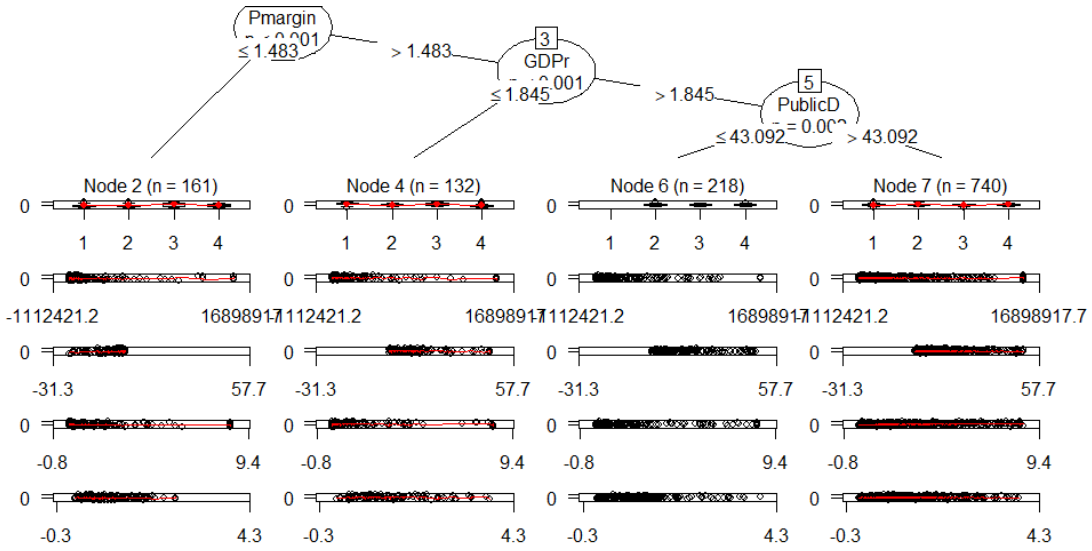


Figura 4: Modelo lineal generalizado combinado con mob

Se observa como, en este caso ha realizado particiones del conjunto escogiendo las variables, *Pmargin, GDPr y PublicD.*

Para que sirva como ejemplo, vamos a observar la salida de `mob` en su primer particionamiento de este modelo:

```
Fluctuation tests of splitting variables:
             Country        Indus        St      AsstT      Pmargin
statistic 37.2223077 2.004729e+02 30.541556 24.8582013 1.121433e+02
p.value    0.5537236 5.135513e-03  0.106365  0.5485383 5.203040e-18
                  ROE        SolvR   FinancPL          Tax
statistic 7.673714e+01 5.226500e+01 24.7717486 38.181099907
p.value   1.609647e-10 1.312603e-05  0.5589344  0.005413061
               Rliquid    Gearing     TobinsQ         GDPr
statistic 43.165388927 29.6861590 39.17076240 1.054387e+02
p.value    0.000683636  0.1456573  0.00361335 1.433526e-16
                Inflat    BudgetB      PublicD     Unemploy
statistic 5.058108e+01 22.4221193 1.046036e+02 7.566241e+01
p.value   2.766295e-05  0.8274277 2.163825e-16 2.682409e-10
              CurrentB       TradeB
statistic 5.148568e+01 4.884072e+01
p.value   1.854712e-05 5.940896e-05

Best splitting variable: Pmargin
Perform split? yes
Node properties:
Pmargin <= 1.483; criterion = 1, statistic = 200.473
```

R-Output 2: Salida de la función mob con verbose = TRUE

de tal forma que el algoritmo escoge aquel valor con un p-valor menor de entre los posibles asociado al test de fluctuación de las variables de split, como se explicó en el capítulo teórico, y después, si es menor a 5% (recordemos que en `mob_control` establecimos `alpha = 0.05`) realiza un particionamiento a partir de esa variable.

Con la función `summary(modelo)` podríamos observar los coeficientes asociados a cada modelo, así como algunos indicadores de calidad, para las variables *Country, Asst, St* ,*Pmargin y Rliquid* que han sido las escogidas en este primer caso. El nodo 4 concentra la muestra que cumple, $Pmargin > 1.483$ y $GPDr \leq 1.845$ y contiene 132 observaciones. Los coeficientes del modelo vienen dados por al siguiente tabla:

```
$`4`

Call:
NULL

Weighted Residuals:
    Min      1Q  Median      3Q     Max
-0.9033  0.0000  0.0000  0.0000  0.8411

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  4.852e-01  6.152e-02   7.887 1.37e-12 ***
Country2    -1.637e-01  9.871e-02  -1.658   0.0998 .
Country3     2.577e-02  8.043e-02   0.320   0.7492
Country4    -3.325e-01  6.235e-02  -5.333 4.41e-07 ***
AsstT       -2.828e-08  6.488e-09  -4.358 2.72e-05 ***
Pmargin     -9.311e-04  2.507e-03  -0.371   0.7109
St          -1.360e-02  1.236e-02  -1.101   0.2732
Rliquid     -2.780e-02  3.473e-02  -0.800   0.4250
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.2949 on 124 degrees of freedom
Multiple R-squared:  0.3565,  Adjusted R-squared:  0.3201
F-statistic: 9.813 on 7 and 124 DF,  p-value: 1.077e-09
```

R-Output 3 summary(mob) nodo teminal 4

Los coeficientes de la variable Country, son todos significativos al menos al 10% menos Country 3. En principio, no parece un modelo un mal modelo, aunque el R^2 no es muy elevado. La variable *AsstT* resulta muy significativa.

Con el fin de mejorar el ajuste implementaremos la función `step()` para saber cuáles son las variables que explican mejor la variable respuesta.

Tabla 2: Ajuste de modelos con mob con step() y deviance

Modelos	MSE	Deviance	LogLik	AIC	Tiempo CPU
GLM con mob	0.062	77.658	-36.53	199.061	6.64
GLM sin mob	0.067	83.567	-82.455	228.91	1.57
Logit con mob	1.049	1312.336	-656.168	1448.336	6638.84
Logit sin mob	1.131	1415.89	-707.945	1475.89	7.28

A través de la implementación de `step()` y respecto a la Tabla 1 hemos podido reducir todos los residuos y la Deviance del modelo, pero no el LogLik y el AIC que han aumentado para el GLM con mob.

Esto se debe a que hemos incluido más variables para realizar la regresión, y el criterio `stepwise()` se realiza sobre todo el conjunto y no sobre cada uno de los subconjuntos de cada nodo. El Stepwise sobre cada subconjunto no podría realizarse ya que entonces no tendríamos un modelo global comparable entre nodos.

Para ver las variaciones del modelo podemos utilizar el `objfun=logLik`, manteniendo todo lo demás constante obtenemos:

Tabla 3: Ajuste de modelos con mob con step() y logLik

Modelos	MSE	Deviance	LogLik	AIC	Tiempo CPU
GLM con mob	0.063	79.69	-32.909	303.81	23.49
GLM sin mob	0.066	83.566	-82.455	228.91	1.31
Logit con mob	4.868	6090.71	-3045.355	6274.71	9191.75
Logit sin mob	1.131	1415.89	-707.945	1475.89	6.30

comparándolo con la Tabla 2 podemos ver que apenas han cambiado los modelos al utilizar el negativo de logLik, por lo que continuaremos utilizando la deviance en el valor `objfun.`

Con el fin de reducir el tiempo de ejecución, sobre todo en el modelo Logit con mob, vamos a eliminar la variable *Indus*, ya que al tratarse de una variable categórica la búsqueda del número óptimo de particiones es del orden de $O(2^{C-1})$, siendo C el número total de categorías, por lo que aumenta exponencialmente el tiempo de ejecución.

Tabla 4: Ajuste de modelos con mob con step(), deviance y sin Indus

Modelos	MSE	Deviance	LogLik	AIC	Tiempo CPU
GLM con mob	0.063	78.819	-44.530	183.060	24.340
GLM sin mob	0.069	86.643	-105.066	240.131	0.530
Logit con mob	1.139	1424.504	-712.252	1470.504	18.840
Logit sin mob	1.167	1460.308	-730.154	1482.308	1.350

Vemos como el modelo GLM con mob empeora los resultados sin variar apenas el tiempo de ejecución, por lo que no la quitaremos para ese caso, sin embargo, para el modelo **Logit con mob**, se reduce en gran medida el tiempo de ejecución, además de que en algunos casos causa problemas realizando demasiados particionamientos, al tratarse de una variable con tantas categorías, por lo que optaremos por quedarnos con el modelo Logit con mob sin Indus.

También al observar la salida en R del modelo **GLM con mob** vemos que algunas variables no son significativas en ninguno de los nodos ni para el modelo GLM con mob, como son *Solv R y Rliquid y Unemploy* , por lo tanto, fijaremos el siguiente valor de fórmula:

```
Perf_2017_cuant ~  Country + AsstT + St  |Country + AsstT + St + Indus
+ Pmargin + ROE + SolvR + FinancPL + Tax + Rliquid +Gearing +
TobinsQ +  GDPr +  Inflat +BudgetB + PublicD + Unemploy + CurrentB
+TradeB
```

Script de R. 3

para el modelo Logit con mob, no eliminaremos tantas variables de regresión, pero si, como hemos comentado, la variable Indus del conjunto de variables de partición:

```
Perf_2017_bi ~ Country + Pmargin + BudgetB + PublicD | Country + St +
AsstT + Pmargin + ROE + SolvR + FinancPL + Tax +Rliquid + Gearing +
TobinsQ +  GDPr +  Inflat + BudgetB + PublicD + Unemploy + CurrentB +
TradeB
```

Script de R. 4

se recogen en la siguiente tabla los indicadores principales de calidad de ambos modelos:

Tabla 5: Modelos finales con mob

Modelos	MSE	Deviance	LogLik	AIC	Tiempo CPU
GLM con mob	0.065	81.313	- 20.646	117.2922	3.69
Logit con mob	1.139	1424.504	-712.252	1470.504	18.84

primero analizaremos el modelo Logit con mob. Empezaremos por ver gráficamente el modelo:

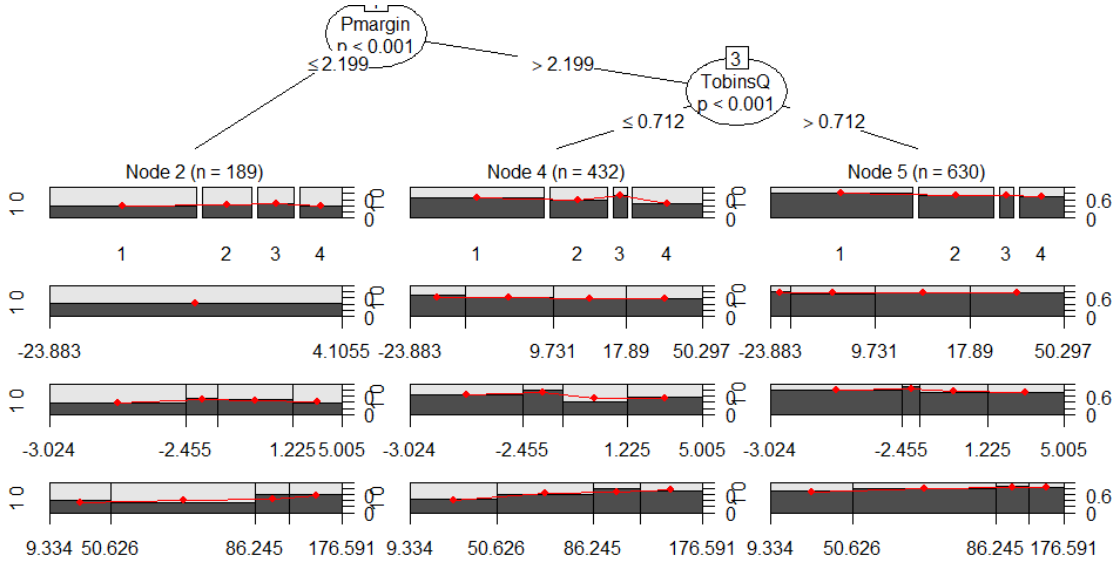


Figura 5: Modelo Logit combinado con mob

Pmargin y *TobinsQ* han sido las variables que han particionado el conjunto. Si observamos el modelo asociado al nodo 5, que es el resultado de particionar el conjunto bajo el criterio Pmargin > 2.199 y TobinsQ > 0.712:

```
Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.1179   0.0000   0.0000   0.6667   1.0092

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept) -0.718359   0.690288  -1.041  0.29803
Country2     1.438564   0.577042   2.493  0.01267 *
Country3     0.371482   0.495921   0.749  0.45381
Country4     0.796532   0.480735   1.657  0.09754 .
Pmargin      0.002832   0.007485   0.378  0.70517
BudgetB     -0.194709   0.089280  -2.181  0.02919 *
PublicD      0.017369   0.006276   2.768  0.00565 **
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 669.93  on 629  degrees of freedom
Residual deviance: 652.15  on 623  degrees of freedom
AIC: 666.15

Number of Fisher Scoring iterations: 4
```

R-Output 4: Salida del nodo 5 del objeto mob

Todas las variables son significativas al menos al 10%, salvo la categoría *Country4* y *Pmargin.* Vemos como todas las variables, al tener coeficientes positivos, afectan de manera directa a la rentabilidad, salvo la variable *BudgetB,* la cual representa el % de déficit de un país, lo cual tiene bastante sentido que tenga una relación indirecta. También, continuando con la interpretación de este nodo, podemos ver las probabilidades asociadas a cada coeficiente en cada nodo utilizando el logaritmo de los odds ratio:

	(Intercept)	Country2	Country3	Country4	Pmargin	BudgetB	PublicD
2	0.079	5.240	1.892	2.684	1.047	0.918	1.022
4	0.078	5.744	3.524	3.395	1.001	0.836	1.028
5	0.488	4.215	1.450	2.218	1.003	0.823	1.018

R-Output 5: Odds ratios del modelo logit con mob

en este caso, para el nodo 5, al aumentar el déficit público, la posibilidad de que se reduzca la rentabilidad es del $1 - 0.823 = 17.7\%$, lo cual es bastante. También vemos como para todos los nodos, la variable *Country2* afecta en gran medida a la rentabilidad, por lo que es de esperar que los países que pertenezcan a esa categoría tengan empresas con rentabilidades altas.

También vamos a calcular otra medida de rendimiento del modelo, a través del porcentaje de casos bien clasificados del conjunto test. Podemos ver el resultado del modelo Logit sin mob y logit con mob a través de la matriz de confusión, y a través del porcentaje de aciertos:

*Tabla 6: Matriz de confusión para el modelo **Logit sin mob***

		Predicción		Porcentaje de acierto
		0	1	71.56%
Observados	0	24	77	
	1	12	200	

*Tabla 7: Matriz de confusión para el modelo **Logit con mob***

		Predicción		Porcentaje de acierto
		0	1	73.16%
Observados	0	41	60	
	1	24	188	

Para realizar predicciones no es tan clara la diferencia de modelos, aunque el modelo Logit con mob clasifica mejor las rentabilidades negativas, y el modelo logit sin mob clasifica algo mejor las rentabilidades positivas. Por lo tanto continúa siendo mejor la aplicación del particionamiento recursivo.

Ahora pasaremos a ver el modelo obtenido en la última tabla, **GLM con mob**. Primero vamos a ver como se distribuye la variable respuesta en los distintos nodos:

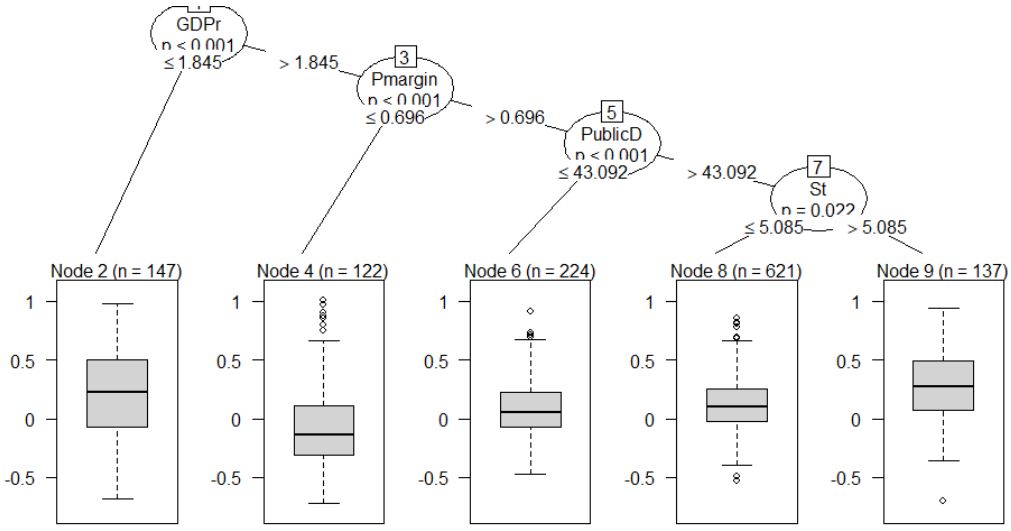


Figura 6: Distribución de la variable respuesta entre los nodos del modelo

El modelo del nodo 2 y el nodo 9 explicarán mejor las empresas con rentabilidades más altas, mientras que las empresas del nodo 4 serán aquellas que tengan unas rentabilidades, por lo general, más bajas.

Podemos ver cuáles han sido las variables y los criterios de partición, así como la distribución de los residuos del modelo asociado a cada variable en cada nodo:

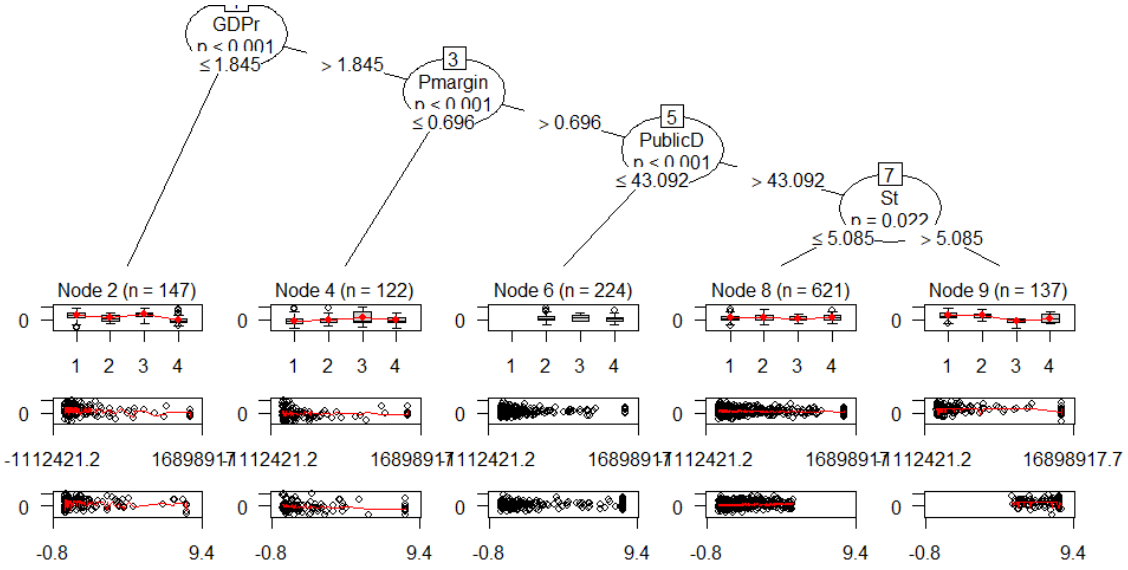


Figura 7: Modelo lineal generalizado combinado con mob

en este caso, las variables por las que se ha particionado han sido, *GDPr, Pmargin, PublicD y St.* Ahora, vamos a pasar a analizar la salida del modelo del nodo 2:

```
$`2`
Call:
NULL
Weighted Residuals:
    Min      1Q  Median      3Q     Max
-0.8761  0.0000  0.0000  0.0000  0.8221

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  4.159e-01  4.630e-02   8.981 1.55e-15 ***
Country2    -1.738e-01  9.723e-02  -1.787 0.076073 .
Country3     8.031e-02  7.504e-02   1.070 0.286359
Country4    -3.248e-01  5.922e-02  -5.485 1.86e-07 ***
AsstT       -2.526e-08  6.469e-09  -3.904 0.000146 ***
St          -2.251e-02  1.219e-02  -1.847 0.066863 .
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.3081 on 141 degrees of freedom
Multiple R-squared:  0.3151, Adjusted R-squared:  0.2908
F-statistic: 12.97 on 5 and 141 DF,  p-value: 2.191e-10
```

R-Output 6: Salida de un nodo del objeto mob

el indicador R cuadrado es bastante mayor que el expuesto en el primer modelo, aunque sigue siendo algo bajo, explicando tan solo el 31,51% de la variabilidad observada de la rentabilidad. Solo la categoría *Country3* tiene un efecto no significativo respecto a la categoría de referencia. También resulta significativa las variable *AsstT*. La rentabilidad tiene una relación inversa respecto a la varianza, y respecto al activo total, lo cual supone un resultado inesperado.

Modelos basados en particionamiento recursivo: Random Forest

Para este apartado, combinaremos particionamiento recursivo insesgado con Random Forest, para conocer cuáles son las variables más relevantes para hacer particiones. Para ello acudiremos al paquete **mobForest** en R y usaremos como referencia [15]. Este paquete se combina funciones del paquete **mob** y **party**, entre otros. La función principal es `mobforest.analysis()`, que toma todos los parámetros de entrada para construir árboles, calcular errores, predicciones y precisión general.

El proceso que va a seguir es el de realizar un número determinado de árboles(se fija ese parámetro en la función) aplicando sobre un conjunto de entrenamiento y también sobre conjunto de testeo. Para cada árbol, seleccionará de manera aleatoria un número de variables que serán candidatas para realizar el particionamiento. El número de variables que seleccione aleatoriamente para cada árbol es un parámetro que podremos fijar en la función.

```
mobforest.analysis(formula, partition_vars, data, mobforest_controls =
mobforest.control(), new_test_data = as.data.frame(matrix(0, 0, 0)),
processors = 1, model = linearModel, family = NULL, prob_cutoff = 0.5,
seed = sample(1:1e+07, 1))
```

Script de R. 5

Siendo `formula` un objeto con variable respuesta y dependientes, igual que en `mob`, escribiendo las variables de particionamiento `partition_vars` como un argumento separado. También tendremos la posibilidad de ajustar la función a través de `mobforest_controls = mobforest.control()`, el cual es muy parecido al `mob_control()` de `mob()`, aunque en este caso tendremos 3 parámetros interesantes para realizar posibles modificaciones:

- `ntree =`: Número de árboles que queramos que nos realice.
- `remplace =`: Para establecer si queremos con o sin reemplazamiento.
- `mtry` = Establece el número de variables que el modelo seleccionará aleatoriamente de las posibles en `partition_vars`.

A la función `mobforest.analysis()` le asignaremos un valor `ntree = 300, mtry = 3` y `remplace = TRUE`.

Una de las aportaciones más interesantes del Random Forest es la posibilidad de medir la importancia de las variables de partición, utilizando la función `varimplot()` sobre el objeto creado. Para el modelo lineal generalizado:

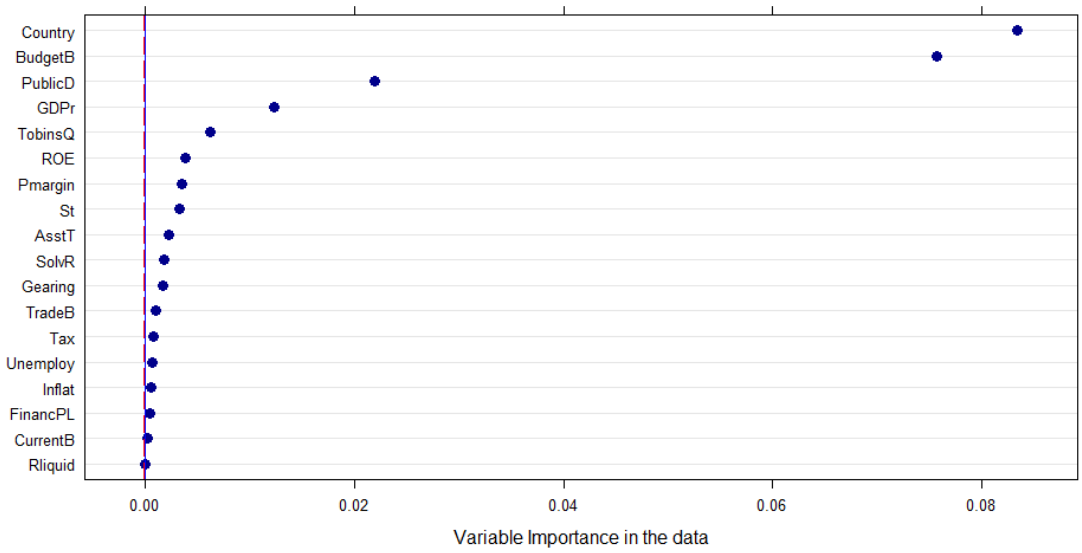


Figura 8: Importancia de las variables para realizar particiones en modelos lineales generalizados

las variables con el punto azul a la derecha de la línea discontinua roja son aquellas que tienen cierta importancia a la hora de realizar particiones. En nuestro caso son principalmente *Country, BudgetB, PublicD, GDPr y TobinsQ.* Destacamos como las variables que más afectan a la hora de realizar particiones son aquellas que influyen por igual a todas, es decir las variables macroeconómicas, lo cual tiene sentido, ya que las rentabilidades de las empresas tienen una evidente relación con la estructura económica del país en el que operan.

Para un modelo logit binomial, asignaremos los mismos valores a la función `mobforest.analysis)`, es decir, `ntree = 300, mtry = 3` y `remplace = TRUE.`

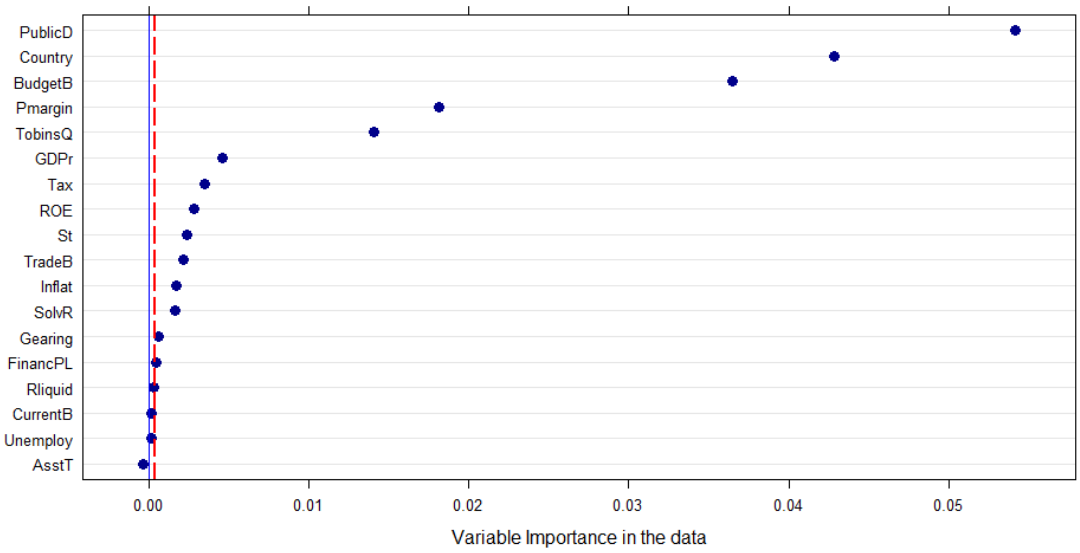


Figura 9: Importancia de las variables para realizar particiones en el modelo logit

Se observa en el gráfico cómo las variables que más particionan para el modelo Logit son similares a las del modelo lineal generalizado, salvo la variable *Pmargin* que en este caso tiene un peso mayor.

Modelo de particionamiento recursivo con modelo logit multinomial: paquete `locClass`

El paquete **`locClass`** nos permite combinar directamente un modelo multinomial original de la librería **`nnet`**, a los `mob` de **`party`**. Simplemente tendremos que cargar la librería y cambiar el valor de `model` = en nuestra función por `multinomModel`. La variable respuesta no se prestaba a hacer muchas subdivisiones cuando se abordó en el análisis exploratorio, por lo que es de esperar que los modelos que obtengamos de la aplicación del modelo multinomial no resulten tan eficientes como las obtenidas para el modelo glm o logit, pero probablemente eso tenga más que ver con la estructura del conjunto de datos, que con la propia validez y utilidad del modelo.

Para abordar este tipo de modelos, cambiaremos la variable respuesta, pasaremos de la variable *Perf_2017_cuant* que es una variable continua, a una variable categórica ordinal. Utilizaremos la variable creada en el capítulo 3: *Perf_2017_ord3*, variable con 3 categorías y *Perf_2017_ord5*, variable con 5 categorías.

Hemos ajustado una serie de modelos incluyendo distintos conjuntos de variables para la regresión y/o partición. En la siguiente tabla se recoge una descripción de cada uno de los modelos ensayados:

Tabla 8: Descripción de los modelos ajustados

Modelos	Descripción
Multinomial sin mob	Unicamente utilizando la función multinom() de la librería nnet.
Modelo 1	Modelo sobre el que hemos pasado un Stepwise, método que ya utilizamos en los apartados anteriores, para que nos oriente sobre que variables pueden ser útiles en la regresión. Hemos incluido todas las variables como posibles para realizar particiones.
Modelo 2	Selección de las variables más relevantes en cuanto a la regresión, eliminando algunas como Country, que nos aportaba información redundante, y eliminando algunas de partición como Indus para ahorrar tiempo de ejecución.
Modelo 3	Selección de 3 variables de regresión y todas en partición salvo Indus
Modelo 4	Selección de las variables de regresión del modelo 2, y escogido únicamente 3 para realizar particiones, que han sido las que nos recomendaba Random Forest.
Modelo 5	Selección únicamente 3 variables de regresión y 3 variables de partición, de acuerdo a los criterios comendados anteriormente.

para el modelo con variable respuesta *Perf_2017_ord5*, es decir la rentabilidad agrupada en 5 categorías hemos obtenido la siguiente tabla:

Tabla 9: Ajuste de modelos multinomiales con 5 categorías con la ***librería locClass***

Modelos	MSE	Deviance	LogLik	AIC	Porcentaje de acierto
Multinomial sin mob	0.1423546	3390	-1695.956	3588.523	35.782
Modelo 1	0.1788369	3345	-1672.	3571.554	35.78
Modelo 2	0.1827103	3450	-1725	3562.061	40.8945
Modelo 3	0.1853214	3450	-1725	3618.91	37.699
Modelo 4	0.1858358	3515	-1758	3663.772	34.504
Modelo 5	0.1893091	3631	-1816	3970.032	36.10

los Modelos etiquetados como 3, 4 y 5 son claramente peores que el resto, tanto en los residuos como en el resto de los indicadores de bondad de ajuste.

Sorprendentemente el modelo que tiene unos residuos más bajos es el modelo multinomial sin mob, aunque en otros indicadores como Deviance elegiríamos el modelo multinomial con todas las variables. Vamos a ver, para que sirva como ejemplo, la representación gráfica del modelo multinomial selección:

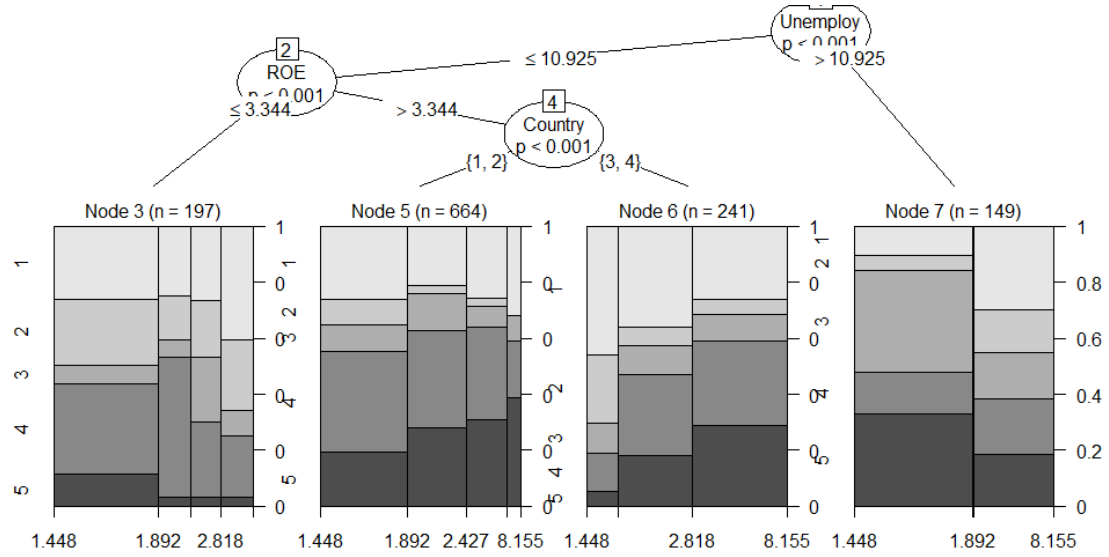


Figura 10: Modelo mob multinomial con 5 categorías. Representación del efecto de la variable GDPr en los nodos terminales.

en este caso hemos optado por representar el efecto de la variable *GDPr* sobre la variable respuesta. El modelo discretiza internamente las variables continuas para mostrar un gráfico de barras agrupado.

En el nodo 3 y en el nodo 7 el efecto de la variable GDPr sobre el rendimiento es negativo , y por el contrario en los nodos 5 y 6 el efecto es positivo.

Los modelos ajustados en los nodos se realizan a través de funciones que hacen uso de la librería **`nnet`**, y la información que ofrecen es más reducida.

Si vemos la matriz de confusión para el modelo con mob y sin mob, junto con sus porcentajes de acierto:

Tabla 10: Matriz de confusión para ***multinomial con 5 sin mob***

		Predicción					Porcentaje de acierto
		1	2	3	4	5	
Observados	1	36	4	4	36	4	35.78%
	2	14	5	2	10	1	
	3	6	2	6	15	15	
	4	29	2	3	54	18	
	5	10	0	5	26	21	

*Tabla 11: Matriz de confusión para **multinomial con 5 con mob***

		Predicción					Porcentaje de acierto
		1	2	3	4	5	
Observados	1	37	3	1	25	8	40.89%
	2	8	8	4	10	2	
	3	7	1	7	12	12	
	4	22	0	6	55	23	
	5	10	0	5	26	21	

el porcentaje de acierto puede parecer bajo, debido posiblemente , a un exceso de segmentación en la variable respuesta que no es ideal para estos datos. Es cierto que sigue siendo mejor que clasificar en una categoría al azar, ya que entonces, hipotéticamente obtendríamos una tasa $1/5$ a cada categoría de ser elegida, y vemos que el porcentaje de acierto de nuestro modelo ronda los $2/3$, lo cual sería significativamente mayor.

Para el modelo con variable respuesta *Perf_2017_ord3*, hemos ajustado la misma batería de modelos que se han descrito en la Tabla 8 cambiando únicamente la variable respuesta.

Los resultados se recogen en la siguiente tabla:

*Tabla 12: Ajuste de modelos multinomiales con 3 categorías con la **librería locClass***

Modelos	MSE	Deviance	LogLik	AIC	Porcentaje de acierto
Multinomial sin mob	0,1841	2320,109	-1160,055	2388,109	60,38339
Modelo 1	0,1394	2155,376	-1077,688	2383,376	60,38339
Modelo 2	0,1489	2289,231	-1144,616	2391,231	64,21725
Modelo 3	0,1518	2274,899	-1137,450	2380,899	61,02236
Modelo 4	0,1535	2307,186	-1153,593	2409,186	55,59105
Modelo 5	0,1583	2360,461	-1180,230	2430,461	55,27157

el mejor modelo es el que incluye todas las variables, eliminando aquellas que no resulten significativas en algunos nodos hojas, o algunas variables redundantes, como *Country*, ya que es

una variable que se formó por la información aportadas por variables como *GDPr* o *BudgetB* en el paso 3 del apartado exploratorio.

Si observamos el modelo Multinomial 2:

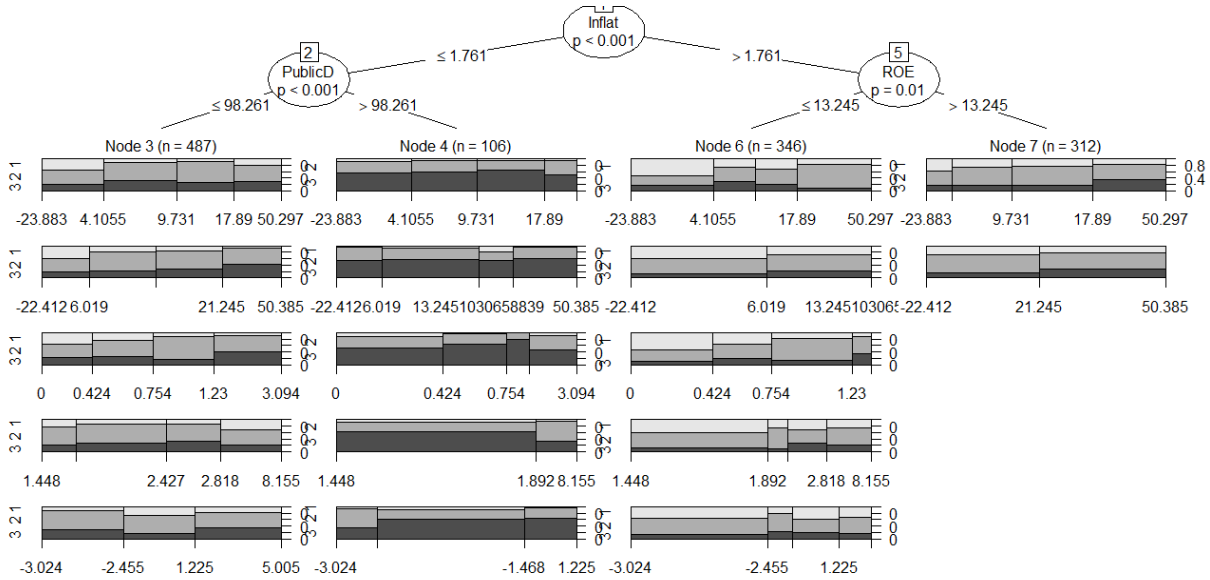


Figura 11: Modelo multinomial con 3 categorías combinado con mob

La mayoría de las observaciones de la categoría 2 están en el nodo 3, mientras que las de la categoría 1 se agrupan en el nodo 4, y en el nodo 6 y nodo 7 se encuentran algo más equilibradas entre la categoría 1 y 2. También vemos como las variables de partición son algunas de las principales que obtuvimos a través del análisis Random forest.

También vamos a representar la matriz de confusión del modelo seleccionado con mob, y el mismo sin mob, junto con sus respectivos porcentajes de acierto:

Tabla 13: Matriz de confusión modelo ***multinomial con 3 categorías sin mob***

		Predicción			Porcentaje de acierto
		1	2	2	
Observados	1	21	37	6	60.38%
	2	18	148	9	
	3	5	49	20	

Tabla 14: Matriz de confusión modelo ***multinomial con 3 categorías con mob***

		Predicción			Porcentaje de acierto
		1	2	2	
Observados	1	20	42	2	64.21%
	2	10	160	5	
	3	4	49	21	

aunque no se aprecian grandes diferencias, sí que vemos que el porcentaje de acierto es mayor sin realizar el mob. Esto puede ser debido a que el modelo multinomial sin mob tiene seleccionadas las variables por Stepwise, mientras que para el modelo con mob, esto no es posible, ya que sería necesario realizarlo para cada nodo por separado, lo que nos llevaría a tener modelos con diferentes variables, y por lo tanto, no comparables.

En este caso, no es tan claro que el particionamiento recursivo combinado con modelos multinomiales mejore el ajuste del conjunto. Esto se debe a que la variable respuesta no tiene una estructura de grupos clara, que dificulta la definición de categorías. Sería interesante aplicar modelos multinomiales con particionamiento recursivo a un conjunto de datos con una estructura en categorías más definida.

3.4 Conclusiones

Las técnicas de particionamiento recursivo nos han permitido aumentar la calidad de los modelos más clásicos como el modelo lineal generalizado o el Logit. Por supuesto, este tipo de técnicas no actúan como sustitutivos a estos, sino que, combinados, permiten ajustar modelos más eficientes de acuerdo con cada una de las regiones del particionamiento que definen.

El conjunto que aquí se ha trabajado es un conjunto de datos complejo, difícil de predecir y al que le afectan muchas más variables de las que el conjunto contiene, lo cual se ha traducido en ruido en nuestros modelos.

En lo que se refiere a la selección de variables de particionamiento, hemos visto como no es muy recomendable incluir variables con muchas categorías, ya que eso causa que aumente exponencialmente el tiempo de ejecución y no necesariamente la calidad del modelo resultante.

Respecto al estudio de la naturaleza de la variable respuesta, y su posible categorización, se ha comprobado que es de utilidad realizar un análisis clúster previo, para comprobar si existen estructuras de agrupación interna que permita disponer de grupos de empresas más homogéneos.

Hemos visto que este tipo de modelos de particionamiento pueden combinarse con otros predictores más actuales, como son los métodos de consenso Random Forest. Sobre nuestro conjunto de datos hemos obtenido que la importancia de las variables de particionamiento coincide tanto en el modelo lineal como en el modelo de regresión logística.

Se puede concluir que las variables macroeconómicas como el déficit público (*BudgetB*) o la tasa de desempleo (*Unemploy*) han sido las más adecuadas para segmentar el conjunto de empresas. Estos resultados son coherentes con los conceptos de riesgo sistemático y diversificación de carteras.

Cuando se ha tratado de predecir la rentabilidad dentro de cada nodo destacan las variables contables, particulares de cada empresa. El activo total (*AsstT*), la desviación típica como indicador de volatilidad (*St*) y el margen sobre beneficio(*Pmargin*) han resultado ser las más significativas.

El objetivo del estudio era la segmentación de las empresas a través de su rentabilidad, para así conseguir unos modelos más eficientes, y efectivamente, las técnicas de particionamiento recursivo basadas en modelos se han revelado como una opción válida, flexible e interesante para alcanzar este objetivo.

3.5 Líneas futuras

Los modelos basados en particionamiento recursivo tienen un gran potencial de trabajo, así como sus diversas implementaciones a distintos conjuntos de datos. Sería muy interesante ver como responden estos modelos ante un conjunto con una estructura categórica más marcada en la variable respuesta, y unas variables dependientes que nos permitiesen aprovechar el potencial del particionamiento recursivo.

También la implementación de distintos modelos en los nodos terminales, como por ejemplo redes neuronales, puede ser un interesante campo de trabajo. Algunas librerías de R han comenzado a implementar modelos de ajuste alternativos para los nodos terminales, aunque queda mucho trabajo por hacer, ya que la diversidad de modelos disponibles es limitada.

Las opciones en la representación gráfica de modelos de particionamiento son muy amplias en la librería `party`, aun así, cabría la posibilidad de representar únicamente un nodo terminal del modelo, o mejorar la estética del gráfico combinándolo con alguna librería ya existente.

La librería `mobForest` abre la posibilidad de combinar particionamiento recursivo y modelos predictivos basados en consenso. Sería muy interesante ver el comportamiento de estos modelos combinados con modelos del tipo boosting. También sería necesario añadir la opción de que ajustase modelos multinomiales o similares.

La selección óptima de parámetros que afectan al funcionamiento interno modelo, contenidos principalmente en la función mob_control(), son mucho más importante de lo que a priori pudiera parecer, y hubiera requerido un estudio de simulación que excedía el objetivo del trabajo.

En lo que al conjunto de datos se refiere, podría aumentarse la muestra seleccionando observaciones de diversos años, y comparando el ajuste del modelo y las predicciones a lo largo del tiempo. También sería posible incluir o excluir variables del actual conjunto con el fin de enriquecer el análisis. En lo que se refiere a la selección de variables para generar un conjunto, el conocimiento experto es relevante para conocer la relación teórica entre las magnitudes que analizan.

Bibliografía

[1] Power D. (2008) *Decision Support Systems: A Historical Overview. In: Handbook on Decision Support Systems 1*. International Handbooks Information System. Springer, Berlin, Heidelberg.

[2] Bishop, C. M. (2006). *Pattern recognition and machine learning*. New York: Springer.

[3] Agresti, A. (1996). *An introduction to categorical data analysis*. New York: Wiley. ISBN: 0471113387 9780471113386

[4] Loh, Wei-Yin. (2014). *Fifty Year of Classification and Regression Trees. International Statistical Review.*

[5] Morgan, J.N. & Sonquist, J.A. (1963). *Problems in the analysis of survey data, and a proposal*. J. Amer. Statist. Assoc.,58, 415–434.

[6] Breiman, L., Friedman, J.H., Olshen, R.A. & Stone, C.J. (1984*). Classification and Regression Trees*. Belmont: Wadsworth.

[7] Aluja T, Nafria E. (1998b) *Generalised impurity measures and data diagnostics in decision trees. Visualising Categorical Data*. ed. Jörg Blasius and M. G reenacre. Academic Press.

[8] Schlosser L, Hothorn T, Zeileis A (2019). *The Power of Unbiased Recursive Partitioning: A Unifying View of CTree, MOB, and GUIDE*. arXiv:1906.10179, arXiv.org E-Print Archive. https://arXiv.org/abs/1906.10179

[9] Breiman, L. (2001). *Random forests*. Mach. Learn., 45, 5–32.

[10] Loh, W.-Y. (2002). *Regression trees with unbiased variable selection and interaction detection*. Stat. Sinica, 12, 361–386

[11] Hothorn, T., Hornik, K. & Zeileis, A. (2006b). *Unbiased recursive partitioning: a conditional inference framework.* J. Comput. Graph. Stat., 15, 651–674.

[12] Zeileis A, Hothorn T, Hornik K (2008). *Model-Based Recursive Partitioning. Journal of Computational and Graphical Statistics*, 17(2), 492–514.

[13] White, H. (1994), *Estimation, Inference and Specification Analysis*, Cambridge University Press.

[14] Zeileis A., Hornik K. (2007), *Generalized M-Fluctuation Tests for Parameter Instability*, Statistica Neerlandica, 61, 488–508.

[15] Gregory C. Chow (1960). «*Tests of Equality Between Sets of Coefficients in Two Linear Regressions*». Econometrica 28 (3): 591-605

[16] Garge, Nikhil & Bobashev, Georgiy & Eggleston, Barry. (2013*). Random forest methodology for model-based recursive partitioning: The mobForest package for R.* BMC bioinformatics. 14. 125. 10.1186/1471-2105-14-125.

[17] AMADEUS : Una base de datos con información financiera comparable de compañías públicas y privadas en Europa (2010). Bureau van Dijk Electronic Publishing.

[18] James, G., Witten, D., Hastie, T., & Tibshirani, R. (2013). *An introduction to statistical learning: With applications in R.*

[19] World Bank Open Data | Data. (s. f.). Recuperado de https://data.worldbank.org

[20] Zeileis, Achim & Hothorn, Torsten & Hornik, Kurt. (2010). *party with the mob: Model-based Recursive Partitioning in R.* Compute.

Glosario de Términos

Todas las definiciones de las variables que incluye el conjunto de datos han sido extraídas de la propia herramienta Amadeus, habiendo acudido también a [18], para completar algunas definiciones.

- *Country:* País en el que cotiza la empresa. Todas las empresas corresponden a empresas europeas.
- *Industry:* Clasificación según la nomenclatura estadística de actividades económicas de la Comunidad Europea (NACE.
- *Perf_2017:* Tasa de variación del precio de la acción entre 2016 y 2017.
- *St:* Desviación típica durante 2017 , calculada como se comenta en 4. **Ajuste y cálculo de variables.**
- *EBITDA* (Beneficio operativo, BAII): Ingresos operativos totales (Ventas netas + Otros ingresos operativos + Variaciones de stock). Las cifras no incluyen IVA. Pueden existir diferencias locales con respecto a los impuestos especiales y pagos obligatorios similares para el mercado específico de las industrias de tabaco y bebidas alcohólicas.
- *AsstT* (Activos totales): Activo corriente + Activo no corriente.
- *Cashf* (Flujo de caja)*:* recursos líquidos generados por la empresa en 2017, medidos como entradas y salidas de efectivo.
- *PL* (Pérdidas / Ganancias)*:* Margen operativo + Margen financiero.
- *CurrentB* (Ratio Actual): Activo corriente / Pasivo corriente.
- *Pmargin* (Margen de beneficio en %): (Beneficio antes de impuestos / Ingresos operativos) * 100.
- *ROE* (Return on Equity)*:* Beneficio neto / Fondos propios.
- *Solv* (Ratio de solvencia basado en activos): Fondos propios / Activos totales.
- *RtdoF*(Ganancias /Pérdidas financieras)*:* Resultado de las actividades financieras de la Compañía (Ingresos financieros-Gastos financieros).
- *Tax* (Impuestos): Todos los impuestos relacionados con el período contable (pagado, acumulado o diferido).
- *ROA* (Retorno en base a los activos): (Ingresos netos / Activos totales) * 100.
- *Rliquid* (Ratio de liquidez)*:* (Activos corrientes - Existencias) / Pasivos corrientes.
- *SolvR* (Ratio de solvencia basado en liquidez)*:* (Fondos propios / Activo total) * 100.
- *Gearing* (Apalancamiento)*:* (Pasivo no corriente + Préstamos) / Fondos propios) * 100.
- *CE/OperatR* (Coste por empleado / Margen Operativo).

- *EaringY* (Rendimiento de Ganancias)*:* Es el cociente entre el beneficio de la empresa entre el precio de la acción y se interpreta como la proporción de las ganancias anuales por acción de una empresa como un porcentaje del precio de mercado por acción.
- *TobinsQ* (Q de Tobin): La Q de Tobin es una medida de los activos de una empresa en relación con el valor de mercado de una empresa. La definición es capitalización de mercado dividida por activos totales. Cuando La Q de Tobin está entre 0 y 1, reemplazar los activos de una empresa cuesta más de lo que vale la empresa. Una Q de Tobin superior a 1 significa que la empresa vale más que el costo de sus activos. Dado que la premisa de Tobin es que las empresas deberían valer lo que valen sus activos, cualquier valor superior a 1 indica teóricamente que una empresa está sobrevalorada.
- *GDPr* (Crecimiento PIB %): Tasa de crecimiento porcentual anual del PIB a precios de mercado en moneda local constante. El PIB es la suma del valor agregado bruto de todos los productores residentes en la economía más los impuestos sobre los productos y menos las subvenciones no incluidas en el valor de los productos.
- *Inflat* (Inflación %): La inflación medida por el IPC refleja el cambio porcentual anual en el coste para el consumidor promedio de adquirir una cesta de bienes y servicios que pueden fijarse o cambiarse en intervalos específicos, en este caso, anualmente.
- *BudgetB* (Balance presupuestario/ Déficit en % de PIB) *:* Mide la diferencia entre los gastos y los ingresos de un país, dividido entre el PIB.
- *PublicD* (Deuda pública en % de PIB) *:* Medido como el stock total de obligaciones contractuales directas de plazo fijo del gobierno con otros pendientes de pago. Incluye pasivos nacionales y extranjeros, como depósitos en moneda y dinero, valores distintos de acciones y préstamos. Es la cantidad bruta de pasivos gubernamentales reducida por la cantidad de capital y derivados financieros mantenidos por el gobierno, divido entre el PIB del país.
- *Unemploy (En % de fuerza laboral):* El desempleo se refiere a la parte de la fuerza laboral que no tiene trabajo pero que está disponible y busca empleo. Las definiciones de fuerza laboral y desempleo pueden diferir según el país.
- *CurrentB* (Saldo por cuenta corriente en % de PIB): El saldo de la cuenta corriente es la suma de las exportaciones netas de bienes y servicios, el ingreso primario neto y el ingreso secundario neto, divididos entre PIB del país.
- *TradeB* (Balanza comercial en % de PIB) *:* Se obtiene de calcular la diferencia entre exportaciones de bienes y servicios e importaciones de bienes y servicios divididos entre el PIB del país.

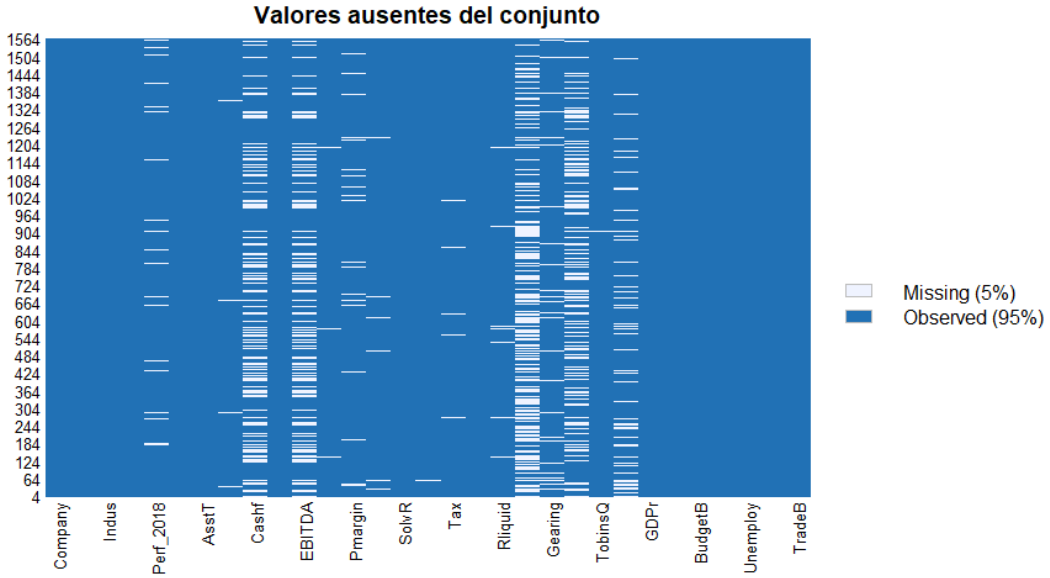
Anexo

1. Descripción del tipo de variables

Este análisis es necesario para comprobar que nuestros datos se han cargado correctamente, y ver si es necesaria la modificación de alguna variable. Utilizaremos el comando `glimse`(), vemos como tenemos que cambiar el valor <chr> de las variables *Indus* y *Country* por factor, para que asocie esos valores a distintos grupos. También algunas variables numéricas han sido calificadas como caracteres, por lo que las cambiaremos a valores numéricos, y también realizaremos un redondeo de 3 cifras.

2. Análisis de datos ausentes

En este apartado utilizaremos la librería *Amelia* para la representación gráfica de los valores ausentes.



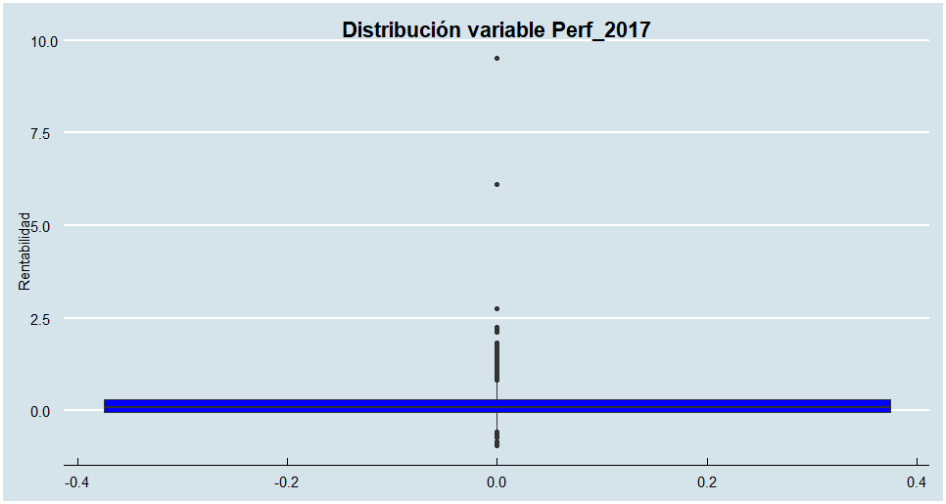
A través de la visualización, y calculando la proporción de valores ausentes, podemos ver que existen 5 variables (*CashF, EBITDA, Rsolv, CE/OperatR* y *EaringY)* con un porcentaje de datos ausentes entre el 10-40% por lo que podemos aplicar métodos para la imputación de valores ausentes, pero dado el excesivo porcentaje, se ha optado por su eliminación.

3. Distribución de las variables

En este apartado realizaremos un análisis descriptivo basado en la visualización utilizando el paquete *ggplot2*.

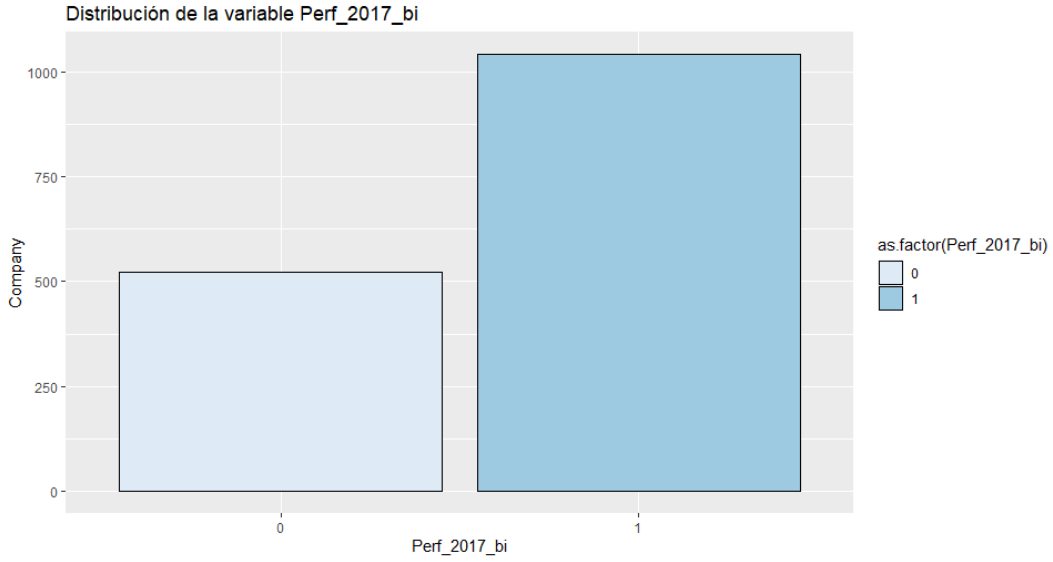
a. Variable respuesta

Nuestra variable respuesta será *Perf_2017* la cual es, originalmente una variable continua expresa la rentabilidad/rendimiento del precio de la acción, calculada como la tasa de variación del precio entre 2017 y 2016. Se distribuye como:



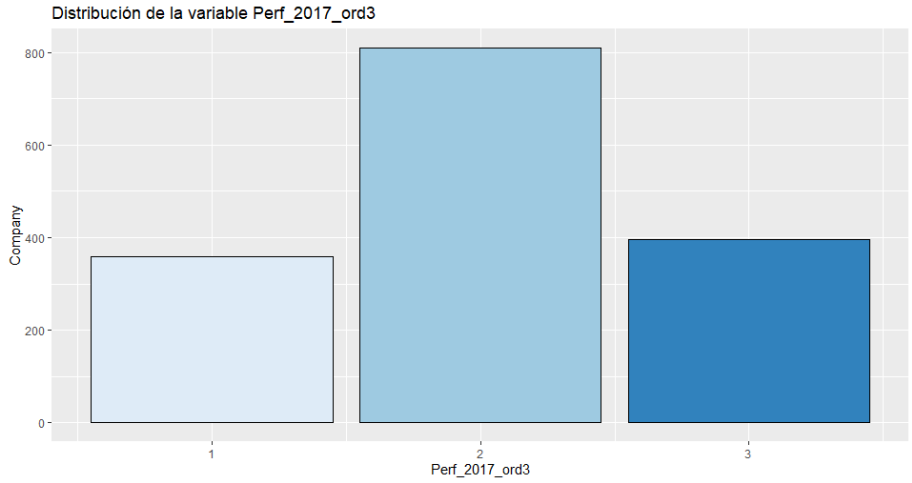
Vemos como tenemos 2 observaciones claramente atípicas, pero el resto parece que se distribuyen normalmente.

Si representamos esta variable como variable binaria siendo 0 las empresas con rentabilidad negativa y 1 las empresas con rentabilidad positiva:

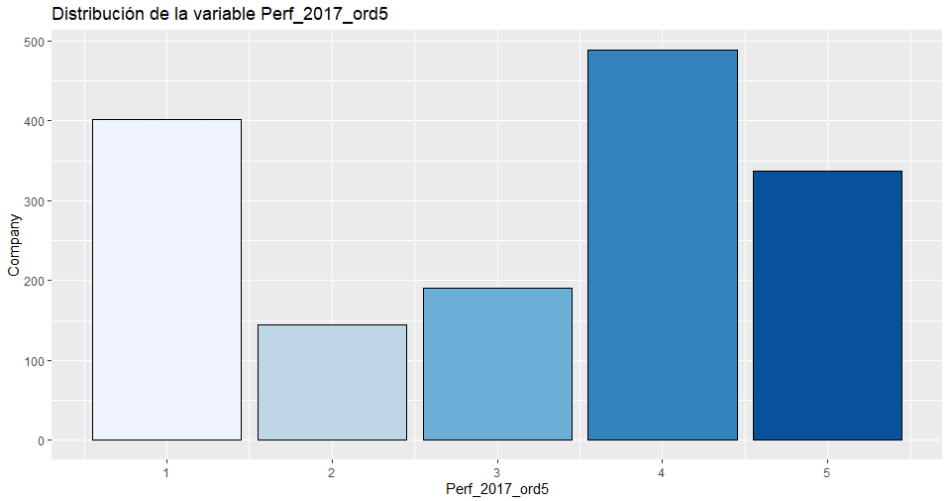


Con la finalidad de implementar modelos más variados, se han creado 2 variables categóricas a partir de Perf_2017_cuant con 3 y 5 categorías cada una. La distribución resultante de cada una de estas variables es:

Perf_2017_ord3



Perf_2017_ord5



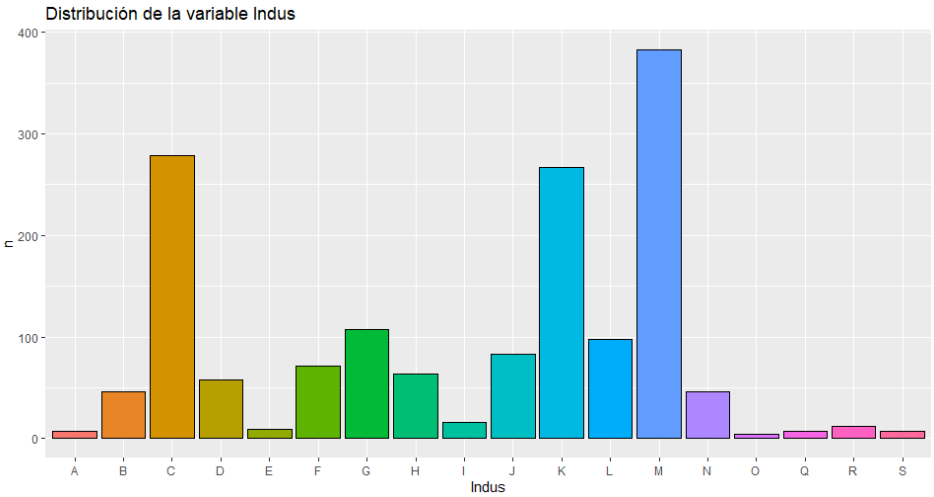
Variables dependientes categóricas:

Indus: Clasificación según la industria. Clasificadas según la nomenclatura estadística de actividades económicas de la Comunidad Europea (NACE) que es el sistema de clasificación de las actividades económicas usado en la Unión Europea, que cuenta con un total de 21 divisiones entre sectores. Esta clasificación nos permitiría hacer subdivisiones para cada categoría, pero para

nuestros datos no parece necesario. En nuestr conjunto final hay empresas de 18 sectores distintos:

Abreviatura	Nace Rev. 2, main section
A	A. Agriculture, forestry and fishing
B	B. Mining and quarrying
C	C. Manufacturing
D	D. Electricity, gas, steam and air conditioning supply
E	E. Water supply; sewerage, waste management and remediation activities
F	F. Construction
G	G. Wholesale and retail trade; repair of motor vehicles and motorcycles
H	H. Transportation and storage
I	I. Accommodation and food service activities
J	J. Information and communication
K	K. Financial and insurance activities
L	L. Real estate activities
M	M. Professional, scientific and technical activities
N	N. Administrative and support service activities
O	O. Public administration and defence; compulsory social security
Q	Q. Human health and social work activities
R	R. Arts, entertainment and recreation
S	S. Other service activities

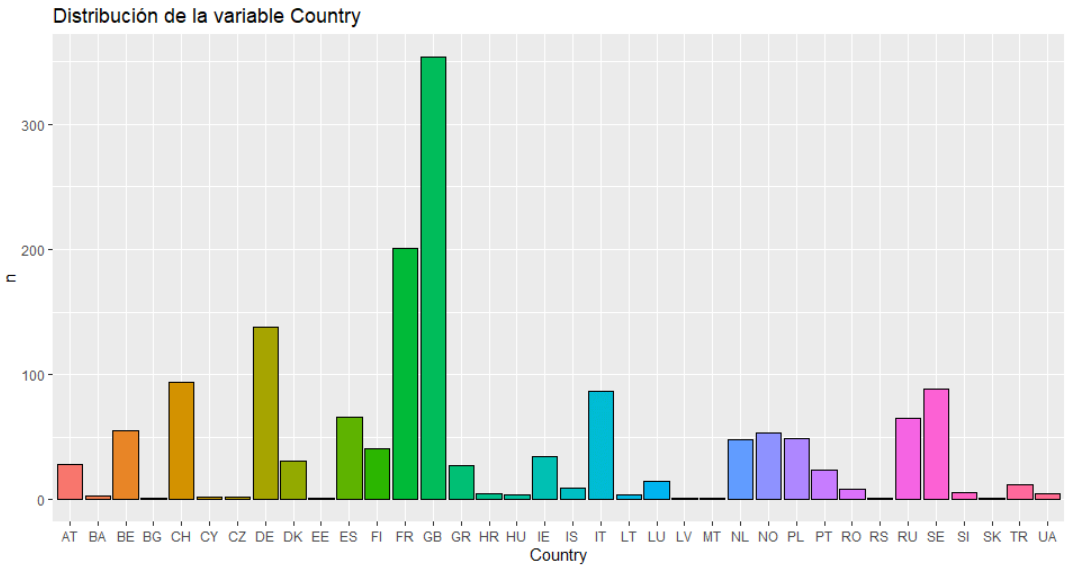
Las cuales se distribuyen:



Country (Country ISO code): País en el que cotiza la empresa. Como ya se ha dicho, todas las empresas corresponden a empresas europeas, y todas son empresas cotizadas, es decir se han excluido del análisis sucursales y empresas no cotizadas. Se utilizará una abreviatura del nombre de los países en base a ISO (International Organization for Standardization) tal que:

ISO code	Country	Nº observaciones	ISO code	Country	Nº observaciones	ISO code	Country	Nº observaciones
AT	Austria	28	GB	United Kingdom	354	NO	Norway	53
BA	Bosnia and Herzegovina	3	GR	Greece	27	PL	Poland	49
BE	Belgium	55	HR	Croatia	5	PT	Portugal	24
BG	Bulgaria	1	HU	Hungary	4	RO	Romania	8
CH	Switzerland	94	IE	Ireland	34	RS	Serbia	1
CY	Cyprus	2	IS	Iceland	9	RU	Russian Federation	65
CZ	Czech Republic	2	IT	Italy	87	SE	Sweden	88
DE	Germany	138	LT	Lithuania	4	SI	Slovenia	6
DK	Denmark	31	LU	Luxembourg	15	SK	Slovakia	1
EE	Estonia	1	LV	Latvia	1	TR	Turkey	12
ES	Spain	66	ME	Montenegro	1	UA	Ukraine	5
FI	Finland	41	MT	Malta	0			
FR	France	201	NL	Netherlands	48			

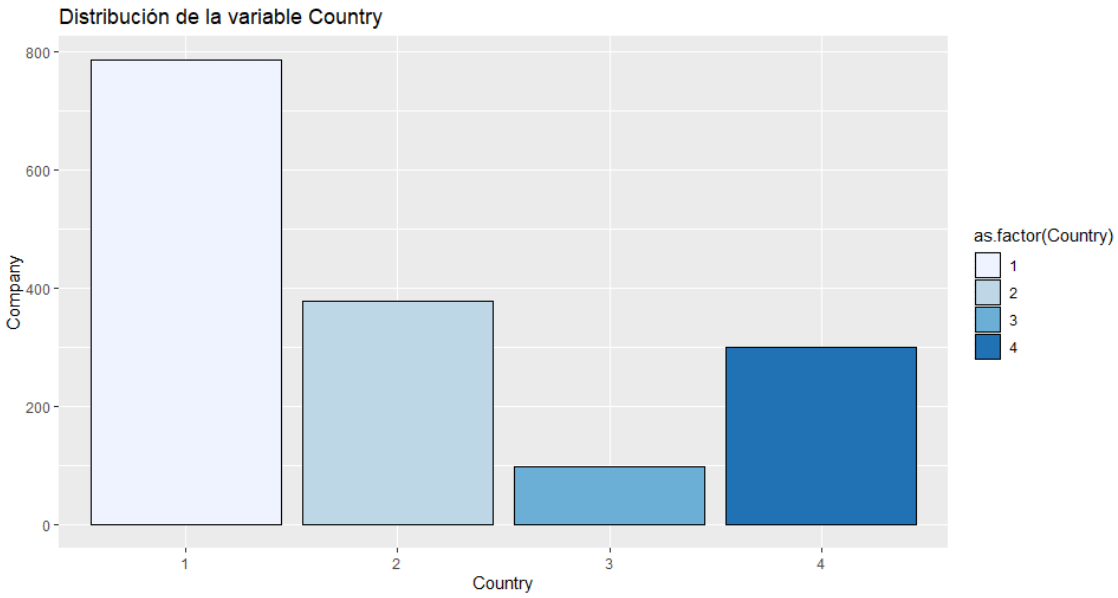
La variable tiene la siguiente distribución de frecuencias



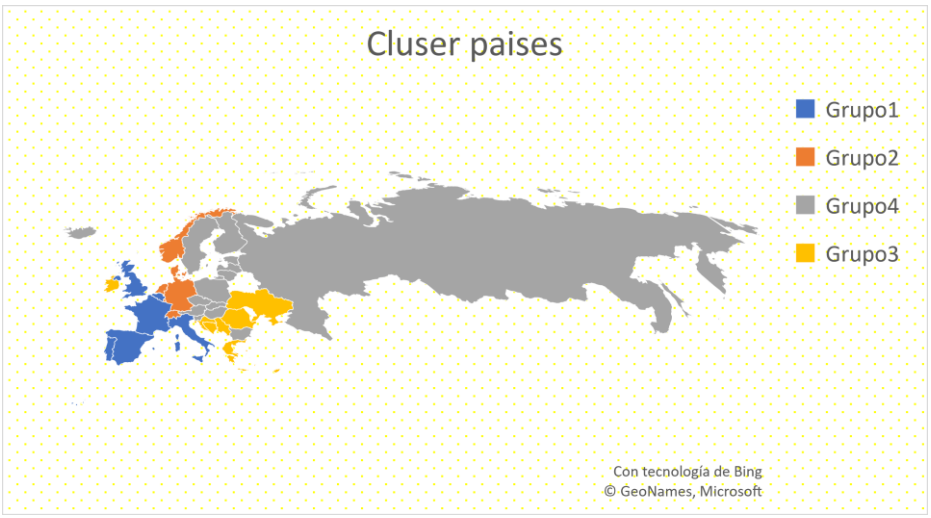
Esta variable posee un gran número de categorías, por lo que con el fin de hacer más eficiente la posterior resolución del modelo, se va a llevar a cabo una agrupación.

3.2.1 Agrupación de la variable Country

Al tratarse la variable que nos indica el país, podríamos llevar a cabo diversas agrupaciones: Por porcentaje de crecimiento, por situación geográfica, por asociación entre países (UE, no UE,…) etc…. . Sin embargo, y ya que disponemos de 6 variables en el conjunto que son indicadores macroeconómicos de estos países, serán las que utilizaremos en la agrupación. Aplicaremos un agrupamiento basado en clustering utilizando la librería *mclust*. De la aplicación de esta técnica obtenemos las siguientes agrupaciónes:



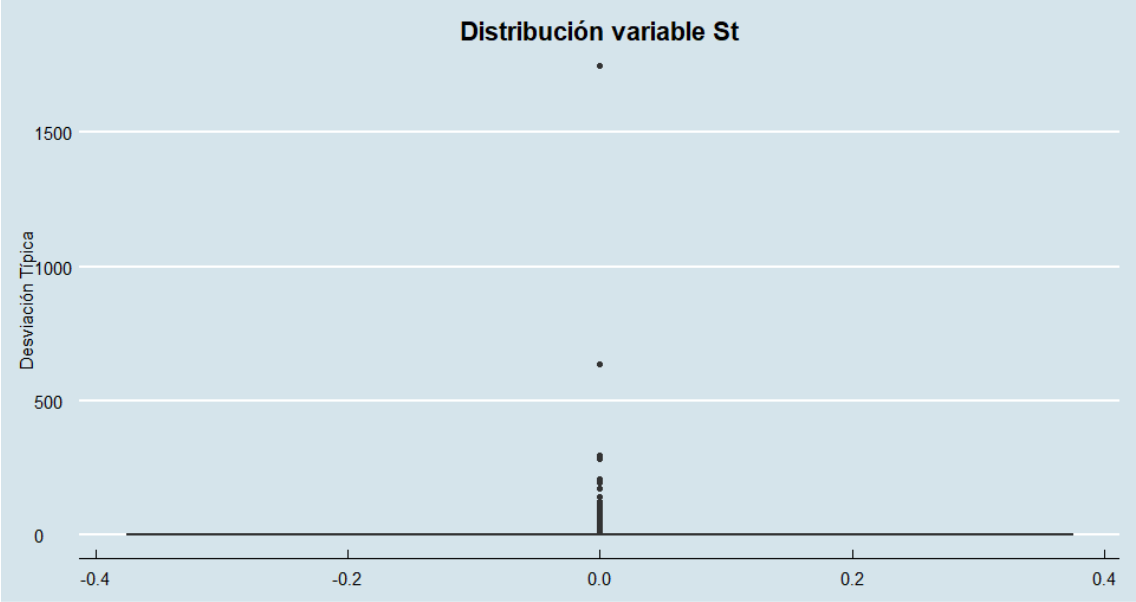
También es interesante visualizar la agrupación sobre un mapa geográfico, resultando:



La agrupación obtenida presenta una alta correlación geográfica, no impuesta a priori en el algoritmo de clustering.

3.1 Variables continuas:

St: Desviación típica durante2017 de cada acción calculada a partir de la cotización mensual de cada empresa durante el año 2017. Nos servirá como medida de volatilidad:



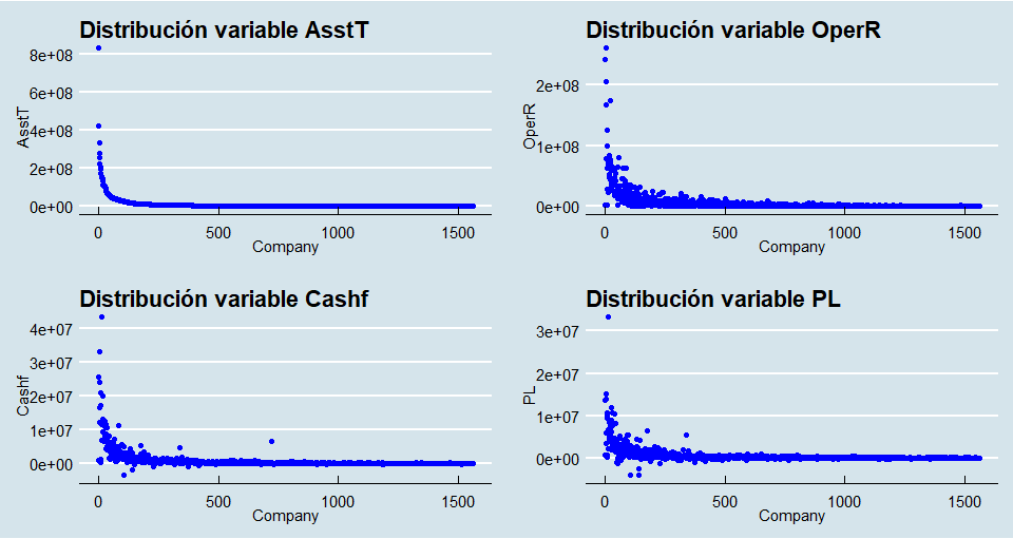

```
summary(fundamentales$St)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
    0.0     0.4     1.2     6.3     3.4  1748.1
```

La presencia de outliers deberá ser tratada en el apartado correspondiente.

Ya que tenemos bastantes variables continuas, vamos a agrupar su exploración entre variables particulares de cada empresa y variables macroeconómicas del país en el que cotiza.

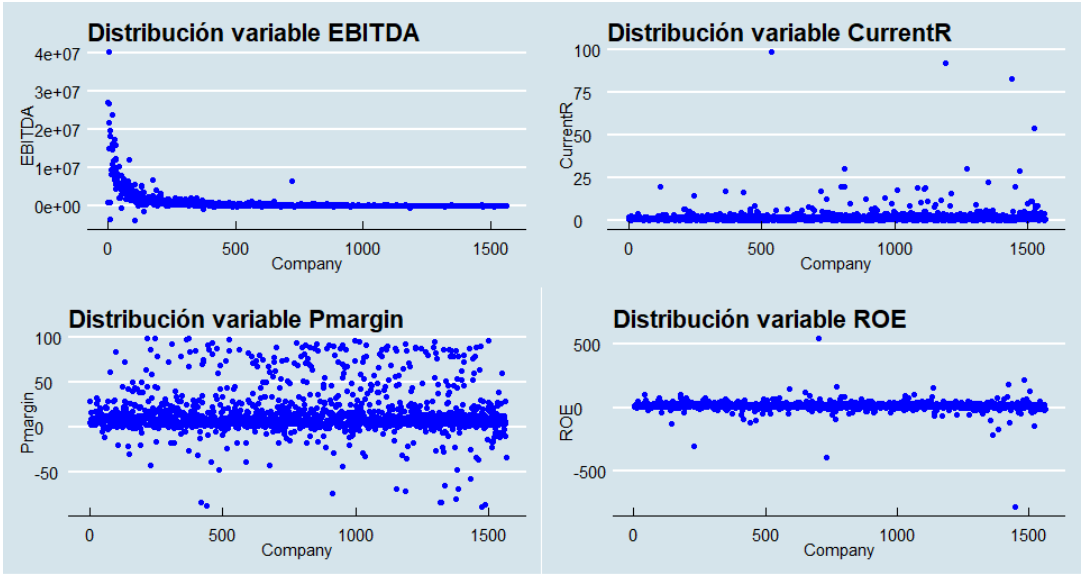
3.2.1 Variables particulares de cada empresa:

En este grupo encontramos las variables : *AsstT, OperR, Cashf, PL, EBITDA, CurrentR, OperR, Pmargin, ROE, SolvR, FinancialPL, Tax, ROA, Rliquid, Rsolv, Gearing , CE/OperarR, TobinsQ, EaringY.*




```
summary(fundamentales_NA[7:10])
AsstT                OperR                Cashf                 PL
Min.   :3.89e+05   Min.   :     -206   Min.   :-3534694   Min.   :-4129000
1st Qu.:7.86e+05   1st Qu.:   406003   1st Qu.:   61634   1st Qu.:   29263
Median :1.70e+06   Median :  1104095   Median :  159328   Median :   94000
Mean   :9.01e+06   Mean   :  5237274   Mean   :  821896   Mean   :  475615
3rd Qu.:5.37e+06   3rd Qu.:  3368087   3rd Qu.:  512425   3rd Qu.:  315745
Max.   :8.36e+08   Max.   :259841669   Max.   :43490771   Max.   :33337705
                    NA's   :21          NA's   :384         NA's   :5
```

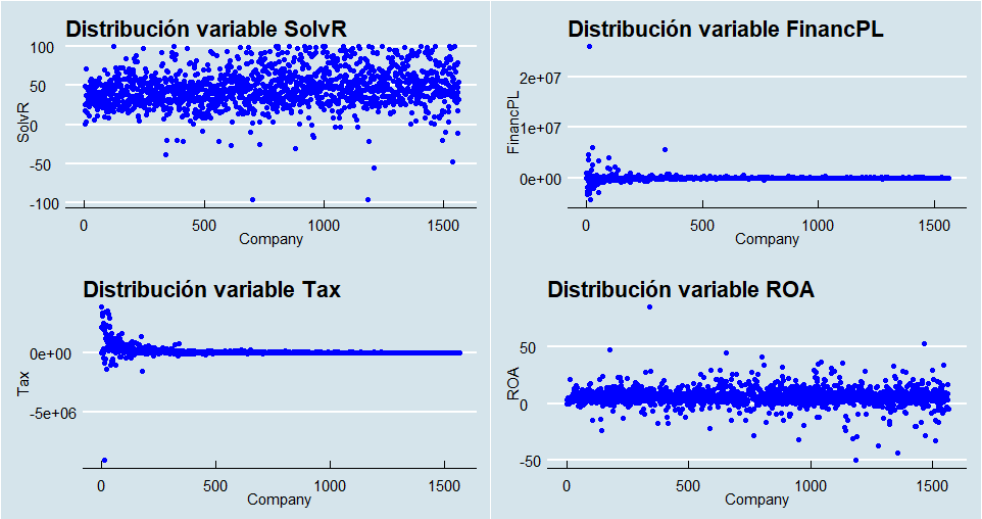
Visualmente se observa cierta correlación entre estas variables, lo cual puede entenderse como que aquellas empresas más grandes tienen un volumen de negocio mayor, por lo que tienen un beneficio operativo o un Cash Flow más alto. También vemos como todas las variables presentan algún outlier . La variable *Cashf* tiene un gran número de NA's, lo cual se tratará en el apartado 4.1.




```
summary(fundamentales_NA[11:14])
EBITDA             CurrentR          Pmargin             ROE
Min.   :-3845853   Min.   : 0.00   Min.   :-88.95   Min.   :-783.1
1st Qu.:   70096   1st Qu.: 0.98   1st Qu.:  3.93   1st Qu.:   6.0
Median :  195951   Median : 1.40   Median :  9.02   Median :  13.2
Mean   :  944607   Mean   : 2.15   Mean   : 15.41   Mean   :  13.2
3rd Qu.:  599356   3rd Qu.: 2.05   3rd Qu.: 18.22   3rd Qu.:  21.2
```

```
Max.   :40321599   Max.   :98.65   Max.   : 98.93   Max.   : 543.2
NA's   :383        NA's   :7       NA's   :78       NA's   :31
```

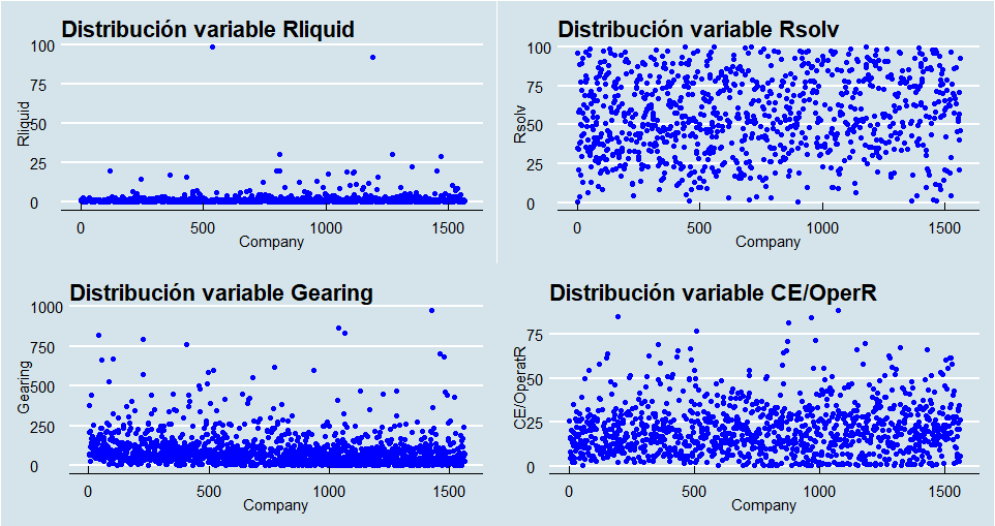
Visualmente la variable EBITDA podría estar correlacionada con las variables del gráfico anterior. El resto de las variables parece que se distribuyen de manera uniforme, aunque basándonos en la distancia del tercer cuartil al máximo y el análisis gráfico, tenemos indicios de presencia de outliers. También vemos como la variable EBITDA tiene prácticamente el mismo número de valores NA's que la variable Cashf, lo cual tiene sentido si atendemos a sus respectivas definiciones[1].




```
summary(fundamentales_NA[15:18])
     SolvR          FinancPL             Tax                ROA
 Min.   :-96.1   Min.   :-4391000   Min.   :-9140238   Min.   :-50.01
 1st Qu.: 32.2   1st Qu.:  -42848   1st Qu.:    3475   1st Qu.:  2.68
 Median : 44.5   Median :   -9055   Median :   17617   Median :  5.63
 Mean   : 46.2   Mean   :  -12662   Mean   :   91068   Mean   :  6.06
 3rd Qu.: 58.1   3rd Qu.:    -586   3rd Qu.:   60952   3rd Qu.:  9.17
 Max.   :100.0   Max.   :26041737   Max.   : 3911612   Max.   : 85.00
                 NA's   :7          NA's   :35         NA's   :5
```

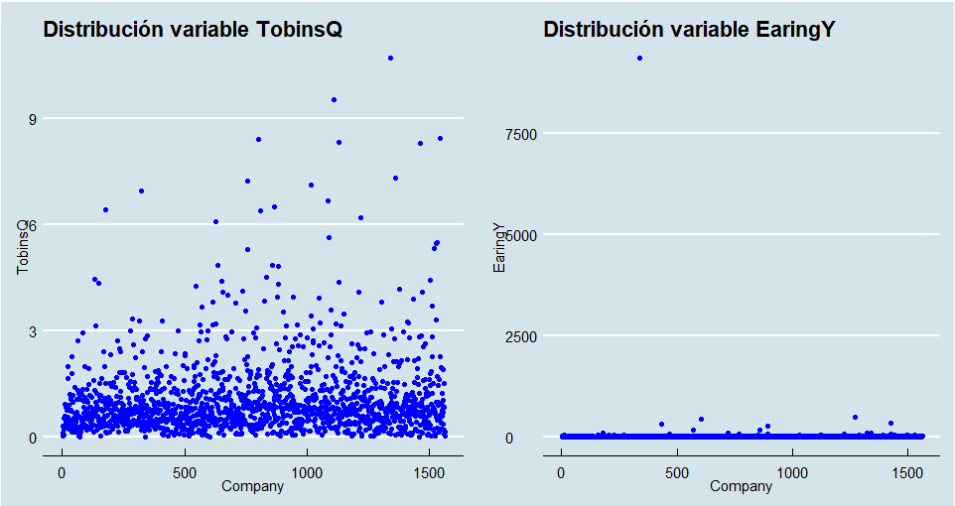
Visualmente parece que se distribuyen de manera uniforme. Tenemos indicios nuevamente de la presencia de outliers, aunque en este caso no tenemos ninguna variable con un gran número de NA's.

[1] Ver Glosario de Términos




```
summary(fundamentales_NA[19:22])
Rliquid           Rsolv           Gearing          CE/OperatR
Min.   : 0.00   Min.   :  0.4   Min.   :  0.0   Min.   : 0.1
1st Qu.: 0.70   1st Qu.: 36.9   1st Qu.: 34.0   1st Qu.:10.4
Median : 1.02   Median : 54.7   Median : 71.1   Median :18.4
Mean   : 1.67   Mean   : 55.4   Mean   : 96.9   Mean   :20.8
3rd Qu.: 1.54   3rd Qu.: 76.2   3rd Qu.:120.7   3rd Qu.:28.0
Max.   :98.65   Max.   :100.0   Max.   :976.7   Max.   :88.9
NA's   :23      NA's   :628     NA's   :104     NA's   :423
```

En las variables *Rliquid* y *Gearing* tenemos indicios de presencia de outliers, aunque el resto de variables se distribuyen de manera uniforme. Destacan la gran cantidad de NA's en las variables *Rsolv* y *CE/OperatR.*. También podemos ver cómo, en este caso, todas las variables tienen valores por encima de 0, lo que concuerda con su definición[2], y nos asegura que no hay errores de codificación para esas variables en ese sentido.



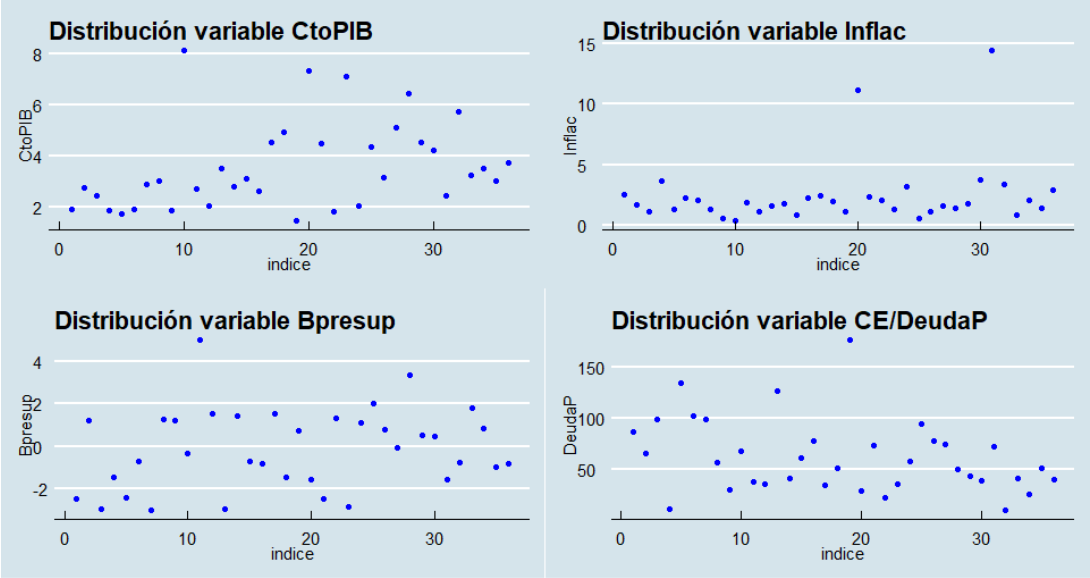
[2] Ver Glosario de Términos.

```
summary(fundamentales_NA[23:24])
TobinsQ          EaringY
Min.   : 0.000   Min.   :   0
1st Qu.: 0.432   1st Qu.:   4
Median : 0.751   Median :   6
Mean   : 1.037   Mean   :  16
3rd Qu.: 1.244   3rd Qu.:   9
Max.   :10.723   Max.   :9373
NA's   :1        NA's   :198
```

Para estas dos últimas variables que nos aportan información relativa a la empresa, vemos como la variable TobinsQ se distribuye de manera bastante uniforme y no tiene valores por debajo de 0, lo cual sería imposible partiendo de su definición. Sin embargo para la variable EaringY tenemos una gran cantidad de valores NA's y un claro valor Outlier.

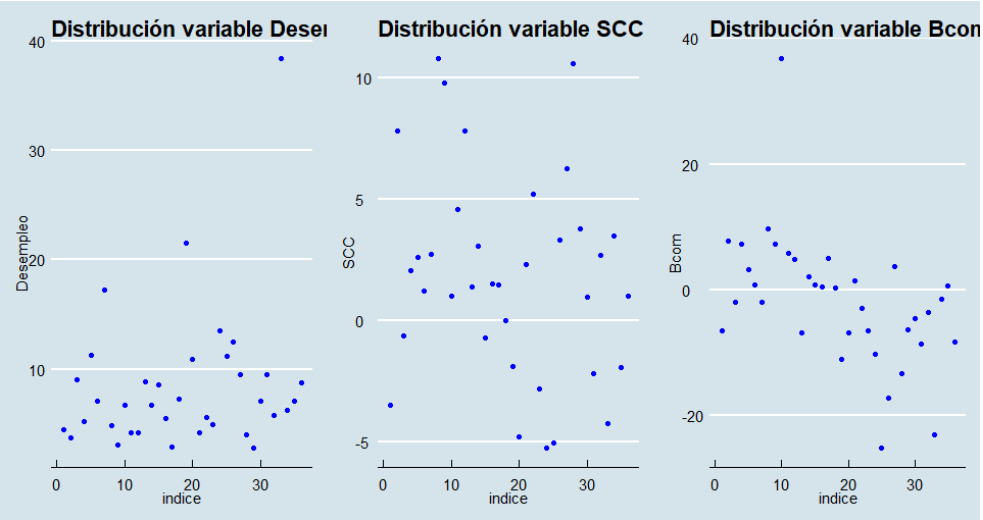
3.2.2 Variables macroeconómicas del país en el que cotiza

En este grupo encontramos las variables*: GDPr, Inflat, BudgetB, PublicD, Unemploy, CurrentB, TradeB.*




```
summary(fundamentales[20:23])
GDPr           Inflat           BudgetB           PublicD
Min.   :1.45   Min.   : 0.341   Min.   :-3.024   Min.   :  9.33
1st Qu.:1.89   1st Qu.: 1.162   1st Qu.:-2.455   1st Qu.: 50.63
Median :2.43   Median : 1.794   Median :-1.468   Median : 86.25
Mean   :2.61   Mean   : 1.918   Mean   :-0.905   Mean   : 74.71
3rd Qu.:2.82   3rd Qu.: 2.579   3rd Qu.: 1.225   3rd Qu.: 98.26
```

En este caso, se observa cierta relación entre las variables, sobre todo en aquellas que pertenecen al grupo 1 tienen empresas con rentabilidad calificada como Muy Alta. No hay valores perdidos ni outliers.




```
summary(fundamentales[24:26])
Unemploy        CurrentB          TradeB
Min.   : 2.81   Min.   :-5.23   Min.   :-25.15
1st Qu.: 4.42   1st Qu.:-1.89   1st Qu.: -6.56
Median : 5.20   Median : 1.23   Median :  0.29
Mean   : 6.90   Mean   : 1.68   Mean   :  0.74
3rd Qu.: 9.07   3rd Qu.: 3.79   3rd Qu.:  5.82
Max.   :38.43   Max.   :10.81   Max.   : 36.90
```

Podemos ver claramente la agrupación que hicimos en el apartado 3.2.1 asocia de manera eficiente aquellos valores semejantes en el mismo grupo. Podemos tener indicios de outliers en las variables *Unemploy y TradeB.*

Anexo código

```
#install.packages("locClassData", repos="http://R-Forge.R-project.org")
#install.packages("locClass", repos="http://R-Forge.R-project.org")
library(dplyr)
library(magrittr)
library(rio)
library(ggplot2)
library(caret)
library(lattice)
library(tidyverse)
library(Amelia)
library(dplyr)
library(readxl)
library(corrplot)
library(nycflights13)
library(grid)
library(party) #mob
library(mob)
library(stats)
library(mobForest) # Random Forest
library(locClass) # Multinomial
library(locClassData)
library(flexmix)
library(nnet)
# Carga de datos
fundamentales_NA<-import("fundamentales_rto.rda")
# Tratamiento de datos perdidos
library(mice)
añadir<-fundamentales_NA[c(1:6)]
columns <- c( "AsstT" ,    "OperR"  ,   "PL"    ,    "CurrentR"
              , "Pmargin"  , "ROE"    ,   "SolvR"   ,  "FinancPL" , "Tax"   ,    "ROA"
,    "Rliquid" ,  "Gearing" ,  "TobinsQ" , "GDPr"
             , "Inflat"  ,  "BudgetB"  , "PublicD"  , "Unemploy" , "CurrentB",
"TradeB" )
imputed_data <- mice(fundamentales_NA[,names(fundamentales_NA) %in% columns],m = 1,
                     maxit = 1, method = "mean",seed = 2018,print = F)
complete.data <- mice::complete(imputed_data)
fundamentales<-data.frame(añadir,complete.data)
# Cluster paises
paises <- read_excel("C:/Users/franc/paises.xlsx",
                     col_types = c("text", "numeric", "numeric",
                                   "numeric", "numeric", "numeric",
                                   "numeric","numeric"))

paises<-data.frame(paises, row.names =paises$Country)
paises<-paises[,-c(1)]
#Componentes principales
paises.PC<-princomp(~. ,data=paises, cor=TRUE, scores=TRUE)
plot(paises.PC$scores[,1:2])
```

```
paises.sc<-scale(paises)
#quete mclust
library(mclust)
mc.res <- Mclust(paises.sc)
fviz_mclust(mc.res, "BIC", palette = "jco") # 4 mejor cluster
fundamentales <- fundamentales %>% mutate(Country = replace(Country, Country ==
"GB"|Country == "FR"|Country == "IT"
                                                            |Country == "BE"|Country ==
"ES"|Country == "PT", "1"))
fundamentales <- fundamentales %>% mutate(Country = replace(Country, Country ==
"DE"|Country == "NL"|Country == "CH"
                                                            |Country == "NO"|Country ==
"DK"|Country == "LU", "2"))
fundamentales <- fundamentales %>% mutate(Country = replace(Country, Country ==
"IE"|Country == "GR"|Country == "TR"
                                                            |Country == "RO"|Country ==
"RS"| Country == "CY"|Country == "HR"
                                                            |Country == "MT"|Country ==
"UA"|Country == "BA", "3"))
fundamentales <- fundamentales %>% mutate(Country = replace(Country, Country ==
"RU"|Country == "SE"|Country == "FI"
                                                            |Country == "AT"|Country ==
"CZ"|Country == "PL"|Country == "HU"
                                                            |Country == "SI"|Country ==
"IS"|Country == "LT"|Country == "EE"
                                                            |Country == "BG"|Country ==
"SK"|Country == "LV" ,"4"))
# Tratamiento de datos atípicos
### Ouliers ###
replace_outliers <- function(x, removeNA = TRUE){
  qrts <- quantile(x, probs = c(0, 0.75), na.rm = removeNA)
  caps <- quantile(x, probs = c(0, .9), na.rm = removeNA)
  iqr <- qrts[2]-qrts[1]
  h <- 1.5 * iqr
  x[x<qrts[1]-h] <- caps[1]
  x[x>qrts[2]+h] <- caps[2]
  x
}
replace_outliers_1 <- function(x, removeNA = TRUE){
  qrts <- quantile(x, probs = c(0.2, 0.8), na.rm = removeNA)
  caps <- quantile(x, probs = c(0.1, 0.90), na.rm = removeNA)
  iqr <- qrts[2]-qrts[1]
  h <- 1.5 * iqr
  x[x<qrts[1]-h] <- caps[1]
  x[x>qrts[2]+h] <- caps[2]
  x
}
# Medias winsortizadas
fundamentales$Perf_2017 <- replace_outliers_1(fundamentales$Perf_2017)
fundamentales$St <- replace_outliers(fundamentales$St)
fundamentales$AsstT <- replace_outliers(fundamentales$AsstT)
fundamentales$OperR <- replace_outliers(fundamentales$OperR)

fundamentales$PL <- replace_outliers_1(fundamentales$PL)
```

```
fundamentales$CurrentR <- replace_outliers(fundamentales$CurrentR)
fundamentales$Pmargin <- replace_outliers_1(fundamentales$Pmargin)
fundamentales$ROE <- replace_outliers_1(fundamentales$ROE)
fundamentales$SolvR <- replace_outliers_1(fundamentales$SolvR)
fundamentales$FinancPL <- replace_outliers_1(fundamentales$FinancPL)
fundamentales$Tax <- replace_outliers_1(fundamentales$Tax)
fundamentales$ROA <- replace_outliers_1(fundamentales$ROA)
fundamentales$Rliquid <- replace_outliers(fundamentales$Rliquid)
fundamentales$Gearing <- replace_outliers(fundamentales$Gearing)
fundamentales$TobinsQ <- replace_outliers(fundamentales$TobinsQ)
# Creación categórica para el rendimiento
fundamentales.PC<-princomp(~. ,data=fundamentales[4], cor=TRUE, scores=TRUE)
fundamentales.sc<-scale(fundamentales[4])
km.res3<-kmeans(fundamentales.sc, 3, nstart=100)
Perf_2017_ord3<-km.res3$cluster
km.res5<-kmeans(fundamentales.sc, 5, nstart=100)
Perf_2017_ord5<-km.res5$cluster



# Análisis de correlación
predict<-fundamentales[c(5)]
empresas<-fundamentales[c(1)]
fundamentales<-fundamentales[,-c(5)]
cualitativas<-fundamentales[2:3]
cuantitativas<-fundamentales[4:25]
St<-fundamentales$St
Perf_2017_cuant<-fundamentales$Perf_2017
cuantitativas<-cuantitativas[,-c(1:2)] #Guardamos y quitamos por si acaso tienen
correlación con otras
# Análisis preliminar
nzv<-nearZeroVar(cuantitativas, saveMetrics= TRUE)
nzv#No tenemos valores con varianza próxima a 0
varcor<-findCorrelation(cor(cuantitativas), cutoff=(0.75))
varcor
#Si elimino aquellas variables que se encuentran correlacionadas ya no tenemos
combinaciones lineales
cuantitativas<-cuantitativas[-varcor]
# Otros ajustes




#Carga de datos
load("fundamentales1.RData")
# División del conjunto en test y entrenamiento
smp_size <- floor(0.8 * nrow(fundamentales))
set.seed(123) # Hacer reproducibles los datos
train_ind <- sample(seq_len(nrow(fundamentales)), size = smp_size)
train <- fundamentales[train_ind, ]
test <- fundamentales[-train_ind, ]
#Modelos principales
ctrl <- mob_control(alpha = 0.05, bonferroni = TRUE, minsplit = 80, trim = 0.1,
```

```
                    breakties = FALSE, parm = NULL, verbose = TRUE, objfun = deviance)
# Este parámetro ctrl va a ser fijo para todos los modelos siguientes
#Glm con mob
system.time(MOB1 <- mob(Perf_2017_cuant~Country + St + AsstT + |Country +
              St + AsstT + Pmargin + ROE + SolvR + FinancPL + Tax +
              Rliquid +Gearing + TobinsQ +  GDPr +  Inflat + BudgetB +
              PublicD + Unemploy + CurrentB + TradeB ,control = ctrl,
            data = train, model = linearModel))
#Indicadores de calidad de ajuste
deviance(MOB1)
logLik(MOB1)
mean(residuals(MOB1)^2)
AIC(MOB1)
#Estos indicadores han sido utilizados en todos los modelos siguientes para medir la
calidad de los mismos
plot(MOB1)
#Glm sin mob
glm1<-glm(Perf_2017_cuant~Country + St + AsstT + ROE  +
            Gearing + TobinsQ + BudgetB + PublicD, data = train)
#Logit con mob
MOB2 <- mob(Perf_2017_bi ~ Country + Pmargin   +
                          BudgetB + PublicD | Country +St + AsstT
                        + Pmargin + ROE + SolvR + FinancPL + Tax +
                          Rliquid +Gearing + TobinsQ +  GDPr +  Inflat +
                          BudgetB + PublicD + Unemploy + CurrentB + TradeB,
                        control = ctrl, data = train,
                        model = multinomModel)
round(exp(coef(MOB2)),3)
#Predicción para Logit con mob
pred_class_MOB2 <- predict(MOB2, newdata = test, out = "class")
mat_conf<-xtabs(~test$Perf_2017_bi+pred_class_MOB2)
sum(diag(mat_conf)/length(test$Perf_2017_bi))
# Logit sin mob
glm2<-glm(Perf_2017_bi~Country + Pmargin   +
            BudgetB + PublicD , data = train, family = "binomial")
#Predicción para Logit sin mob
cats.prob <- predict(glm2, newdata = test, type = "response")
cats.pred = rep("0", dim(test)[1])
cats.pred[cats.prob > .5] = "1"
mat_conf<-xtabs(~test$Perf_2017_bi+cats.pred)
sum(diag(mat_conf)/length(test$Perf_2017_bi))
# Random forest
rfout <- mobforest.analysis(as.formula(Perf_2017_cuant ~ Country + St + AsstT + ROE  +
                                         Gearing + TobinsQ + BudgetB + PublicD ),
c("Country","St", "AsstT",

"Pmargin","ROE","SolvR","FinancPL","Tax",

"Rliquid",  "Gearing","TobinsQ","GDPr","Inflat",

"BudgetB","PublicD","Unemploy","CurrentB",
```

```
"TradeB")
                          ,mobforest_controls = mobforest.control(ntree =150, mtry =
3, replace = TRUE,
                                                                  alpha = 0.05,
bonferroni = TRUE, minsplit = 80, verbose = TRUE)
                          ,data = train,processors = 1, model = linearModel, seed =
1234, new_test_data = test)
varimplot(rfout)
rfout_bi <- mobforest.analysis(as.formula(Perf_2017_bi ~ Country + Pmargin   +
                                        BudgetB + PublicD), c("Country","St",
"AsstT",

"Pmargin","ROE","SolvR","FinancPL","Tax",
                                                                  "Rliquid",
"Gearing","TobinsQ","GDPr","Inflat",

"BudgetB","PublicD","Unemploy","CurrentB",
                                                                  "TradeB")
                            ,mobforest_controls = mobforest.control(ntree =150, mtry
= 3, replace = TRUE,
                                                                      alpha = 0.05,
bonferroni = TRUE, minsplit = 80, verbose = TRUE)
                            ,data = train,processors = 1, model = glinearModel, seed
= 1234, new_test_data = test,family=binomial())
varimplot(rfout_bi)
# Modelos multinomiales con 5 con mob
MOB_multi5_mejor <- mob(Perf_2017_ord5 ~  St + ROE + TobinsQ +
                         GDPr + BudgetB| Country + St + AsstT + Pmargin + ROE + SolvR +
FinancPL + Tax +
                         Rliquid +Gearing + TobinsQ +  GDPr +  Inflat +
                         BudgetB + PublicD + Unemploy + CurrentB + TradeB
                        ,control = ctrl, data = train, trace = FALSE,
                        model = multinomModel)
summary(MOB_multi5_mejor) # Ver relación
plot(MOB_multi5_mejor, tp_args = list(fitmean = FALSE))
round(exp(coef(MOB_multi5_mejor)),3)
pred_class_multinom <- predict(MOB_multi5_mejor, newdata = test, out = "class")
mat_conf_multi<-xtabs(~test$Perf_2017_ord5+pred_class_multinom)
sum(diag(mat_conf_multi)/length(test$Perf_2017_ord5))
summary(MOB_multi3_mejor)
# Multinomial 5 sin mob
multinom5_sinmob<-multinom(Perf_2017_ord5~Country + St + ROE + SolvR +
                            Tax + TobinsQ + GDPr + BudgetB + PublicD + CurrentB +
AsstT, data = train)
pred_class_multinomsin <- predict(multinom5_sinmob, newdata = test, out = "class")
mat_conf_multisin<-xtabs(~test$Perf_2017_ord5+pred_class_multinomsin)
sum(diag(mat_conf_multisin)/length(test$Perf_2017_ord5))

# Modelos multinomiales con 3 con mob
MOB_multi3_mejor <- mob(Perf_2017_ord3 ~  Pmargin + ROE + TobinsQ +
                         GDPr + BudgetB| Country + St + AsstT + Pmargin + ROE + SolvR +
FinancPL + Tax +
                         Rliquid +Gearing + TobinsQ +  GDPr +  Inflat +
                         BudgetB + PublicD + Unemploy + CurrentB + TradeB
                        ,control = ctrl, data = train, trace = FALSE,
```

```
                        model = multinomModel) # Al tener las variables GDPr + BudgetB,
es  redundante incluir Country
plot(MOB_multi3_mejor, tp_args = list(fitmean = FALSE))
pred_class_multinom <- predict(MOB_multi3_mejor, newdata = test, out = "class")
mat_conf_multi<-xtabs(~test$Perf_2017_ord3+pred_class_multinom)
sum(diag(mat_conf_multi)/length(test$Perf_2017_ord3))
# Multinomial sin mob
multinom3_sinmob<-multinom(Perf_2017_ord3~Country + St + AsstT + Pmargin +
                           ROE + SolvR + FinancPL + Gearing + TobinsQ + GDPr + Inflat
+
                           BudgetB + PublicD + Unemploy, data = train)
pred_class_multinomsin <- predict(multinom3_sinmob, newdata = test, out = "class")
# MAatriz de confusión
mat_conf_multisin<-xtabs(~test$Perf_2017_ord3+pred_class_multinomsin)
#Probabilidad de acierto
sum(diag(mat_conf_multisin)/length(test$Perf_2017_ord3))
```